Code Mode Documentation - 2/3

dice, hilo, keno, limbo, mines, plinko, tower, pump

---

system 1 June 10, 2025, 10:00am

Darts

Variables

- `betSize`
  Type: float
  Required: true
- `risk`
  Type: string
  Required: true
  Possible values: easy, medium, hard, expert

lastBet example


```
{
"_id": "XXXX-XXXX-XXXX-XXXX-XXXX",
"_iid": null,
"_rollNumber": 14,
"_nonce": "N/A",
"_active": false,
"_game": "darts",
"_product": "ORIGINALS",
"_win": false,
"_amount": 0,
"_fiatAmount": "₫0.00",
"_payoutMultiplier": 0.4,
"_multiplierResult": 0.4,
"_b2bMultiplier": 0.4,
"_payout": 0,
"_fiatPayout": "₫0.00",
"_currency": "TRX",
"_fiatCurrency": "VND",
"_dateTime": "2025-06-17T05:38:39.627Z",
"_deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
"_state": {
  "distance": 0.47591697498574637,
  "rotation": 0.06237688264809549,
  "difficulty": "medium"
},
"_provider": "originals"
```

```
}
```

Code example

```
// Flat bet on medium difficulty

// resetStats();
// resetSeed();

game = 'darts';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
difficulty = 'medium'

engine.onBetPlaced(async (lastBet) => {

});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
});
```

Dice

Variables

- `betSize`
  Type: float
  Required: true
- `target`
  Type: float
  Required: true
- `betHigh`
  Type: boolean
  Required: true

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 5,
  "nonce": 1027,
  "active": false,
  "game": "dice",
  "win": true,
```

```
  "amount": 1,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 2,
  "b2bMultiplier": 2,
  "payout": 2,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "result": 87.54,
    "target": 50.5,
    "multiplierTarget": 2,
    "condition": "above"
  },
  "clientSeed": "yourClientSeed",
  "serverSeed": "XXX", // Simulation Mode only
  "serverSeedHashed": "XXX" // Live Mode only
}
```

Code example

```
// Basic martingale example with stop on loss streak of 10

// resetStats();
// resetSeed();

game = 'dice';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
betHigh = true;
target = chanceToMultiplier(49.5);

engine.onBetPlaced(async (lastBet) => {
    if (currentStreak === -10) {
        log('Loss streak of 10 reached. Stopping...');
        engine.stop();
    }

    if (lastBet.win) {
        betSize = initialBetSize;
    } else {
        betSize *= 2;
    }
});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
```

Tower

Variables

- `betSize`
  Type: float
  Required: true
- `difficulty`
  Type: string
  Required: true
  Possible values: easy, medium, hard, expert, master
- `tiles`
  Type: array
  Required: true
  Example: `tiles = [0, 0, 1, 1];`
  Each number represents the index of a row. If you have an array with 3 numbers, that means you're betting on 3 rows.
  Be aware, that Stake doesn't validate if the chosen number is valid for the current difficulty! You have to make sure to not use `2` for difficulty `hard`, for example, because there are only 2 columns to choose from and the numbers start with 0.

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 6,
  "nonce": 1027,
  "active": false,
  "game": "tower",
  "win": true,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 1.74,
  "b2bMultiplier": 1.74,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "currentRound": 2,
    "playedRounds": [
      [2, 0, 1],
      [1, 0, 3],
      [2, 0, 1],
      [2, 0, 3],
```

```
      [0, 1, 2],
      [0, 1, 3],
      [3, 2, 1],
      [3, 1, 2],
```

Code example

```
// Always bet on 2 rows and cycle through the difficulties

// resetStats();
// resetSeed();

function randomNumber(min, max) {
    return Math.floor(Math.random() * (max - min + 1) + min);
}

function randomTiles(amount, difficultyIndex) {
    const maxTiles = [4, 3, 2, 3, 4],
        tiles = [];

    while (tiles.length < amount) {
        tiles.push(randomNumber(0, maxTiles[difficultyIndex] - 1));
    }

    return tiles;
}

game = 'tower';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
difficulties = casino === 'SHUFFLE' ? ['EASY', 'MEDIUM', 'HARD', 'EXPERT', 'MAST
difficultyIndex = 0;
difficulty = difficulties[difficultyIndex];
rows = 2;
```

Hilo

Variables

- `betSize`
  Type: float
  Required: true
- `startCard`
  Type: object
  Required: true
  Example: `{rank: "K", suit: "C"}`
  Suit keys: “C” = Clubs, “D” = Diamonds, “H” = Hearts, “S” = Spades

onGameRound return constants

- `HILO_BET_HIGH`
  Type: string
- `HILO_BET_HIGH_EQUAL`
  Type: string
- `HILO_BET_LOW`
  Type: string
- `HILO_BET_LOW_EQUAL`
  Type: string
- `HILO_SKIP`
  Type: string
  Hint: You can skip at maximum 52 cards in one Hilo bet!
- `HILO_BET_EQUAL`
  Type: string
- `HILO_CASHOUT`
  Type: string

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 6,
  "nonce": 1027,
  "active": true,
  "game": "hilo",
  "win": true,
  "amount": 2,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 1.161875,
  "b2bMultiplier": 1.161875,
  "payout": 2.32375,
  "fiatPayout": "$0.00",
  "currency": "trx",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "startCard": {
      "suit": "C",
      "rank": "A"
    },
    "rounds": [
```

```
      {
        "card": {
          "suit": "S",
          "rank": "K"
        },
```

Code example

```
// Increase bet size every 2nd loss, reset on win

// resetStats();
// resetSeed();

game = 'hilo';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
startCard = {rank: "7", suit: "C"};
cashOutAtMultiplier = 3;
increaseOnLoss = 2;

engine.onBetPlaced(async (lastBet) => {
    if (lastBet.win) {
        betSize = initialBetSize;
    }

    if (currentStreak < 0 && currentStreak % -2 === 0) {
        betSize *= increaseOnLoss;
    }
});

engine.onGameRound((currentBet) => {
    // Fetching current card rank, fallback to start card rank, because rounds :
    currentCardRank = currentBet.state.rounds.at(-1)?.card.rank || currentBet.st
    payoutMultiplier = currentBet.state.rounds.at(-1)?.payoutMultiplier || 0;
    skippedCards = currentBet.state.rounds.filter(round => round.guess === 'skip
```

Keno

Variables

- `betSize`
  Type: float
  Required: true
- `risk`
  Type: string
  Required: true
  Possible values: classic, low, medium, high

- `numbers`
  Type: array
  Required: true
  Example: `numbers = [1, 8, 10, 16, 18, 25];`
  Numbers are the range from 0 to 39. At least one number required.

lastBet example

```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 7,
  "nonce": 1027,
  "active": false,
  "game": "keno",
  "win": false,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 0,
  "b2bMultiplier": 0,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "drawnNumbers": [0, 9, 13, 16, 27, 29, 30, 31, 36, 38],
    "selectedNumbers": [7, 9, 11, 17, 19, 22, 26, 31, 33, 35],
    "risk": "medium"
  },
  "clientSeed": "yourClientSeed",
  "serverSeed": "XXX", // Simulation Mode only
  "serverSeedHashed": "XXX" // Live Mode only
}
```

Code example

```
// Add 3 random numbers on win, remove one random number on loss.
// Keep minimum of 1 and a maximum of 10 numbers.

// resetStats();
// resetSeed();

function randomNumbers(amount) {
    // Create array with a number range from 0 to 39
    const numbers = Array.from(Array(40).keys());

    return shuffle(numbers).slice(0, amount);
}
```

```
game = 'keno';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
risk = 'medium';
amount = 10;
numbers = randomNumbers(amount);
minNumbers = 1;
maxNumbers = 10;

engine.onBetPlaced(async (lastBet) => {
    if (lastBet.win) {
        // Add 3 numbers on win, make sure we have 10 numbers max.
        amount = Math.min(maxNumbers, amount + 3);
    } else {
```

Limbo

Variables

- `betSize`
  Type: float
  Required: true
- `target`
  Type: float
  Required: true
  Possible values: 1.00000000000001 - 1000000

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 8,
  "nonce": 1027,
  "active": false,
  "game": "limbo",
  "win": true,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 2,
  "b2bMultiplier": 2,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
```

```
    "result": 39.07902141536203,
    "multiplierTarget": 2,
  },
  "clientSeed": "yourClientSeed",
  "serverSeed": "XXX", // Simulation Mode only
  "serverSeedHashed": "XXX" // Live Mode only
}
```

Code example

```
// Increase target on loss, reset to initial target on win, static bet size

// resetStats();
// resetSeed();

game = 'limbo';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
target = 2;
initialTarget = target;

engine.onBetPlaced(async (lastBet) => {
    if (lastBet.win) {
        target = initialTarget;
    } else {
        target += 1.05;
    }
});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
});
```

Mines

Variables

- `betSize`
  Type: float
  Required: true
- `mines`
  Type: integer
  Required: true
  Possible values: From 1 to 24

- `fields`
  Type: array
  Required: true
  Example: `numbers = [0, 8, 10, 16, 18, 24];`
  Numbers are the range from 0 to 24. At least one number required.

lastBet example

```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 9,
  "nonce": 1027,
  "active": false,
  "game": "mines",
  "win": false,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 0,
  "b2bMultiplier": 0,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "mines": [5, 7, 14],
    "minesCount": 3,
    "rounds": [
      {"field": 2, "payoutMultiplier": 0},
      {"field": 5, "payoutMultiplier": 0},
      {"field": 6, "payoutMultiplier": 0},
      {"field": 7, "payoutMultiplier": 0},
      {"field": 14, "payoutMultiplier": 0},
      {"field": 15, "payoutMultiplier": 0},
      {"field": 16, "payoutMultiplier": 1.125},
```

Code example

```
// Add 3 random numbers on win, remove one random number on loss.
// Keep minimum of 1 and a maximum of 10 numbers.

// resetStats();
// resetSeed();

function randomFields(amount) {
    // Create array with a number range from 0 to 24
    const fields = Array.from(Array(25).keys());

    return shuffle(fields).slice(0, amount);
```

```
}

game = 'mines';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
mines = 3;
amount = 10;
fields = randomFields(amount);
minFields = 1;
maxFields = 10;

engine.onBetPlaced(async (lastBet) => {
    if (lastBet.win) {
        // Add 3 numbers on win, make sure we have 10 numbers max.
        amount = Math.min(maxFields, amount + 3);
    } else {
```

Plinko

Variables

- `betSize`
  Type: float
  Required: true
- `risk`
  Type: string
  Required: true
  Possible values: low, medium, high
- `rows`
  Type: integer
  Required: true
  Numbers in the range from 8 to 16.

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 10,
  "nonce": 1027,
  "active": false,
  "game": "plinko",
  "win": false,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 0.5,
  "b2bMultiplier": 0,
```

```
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "risk": "medium",
    "rows": 16,
    "point": 405.12394637942276,
    "path": ["R", "R", "L", "L", "R", "L", "R", "R", "L", "L", "R", "L", "R", "R
  },
  "clientSeed": "yourClientSeed",
  "serverSeed": "XXX", // Simulation Mode only
  "serverSeedHashed": "XXX" // Live Mode only
}
```

Code example

```
// Basic martingale example with stop on loss streak of 10

// resetStats();
// resetSeed();

game = 'plinko';
betSize = 0.00000000;
initialBetSize = betSize;
risk = 'medium';
rows = 16;

engine.onBetPlaced(async (lastBet) => {
    if (currentStreak === -10) {
        log('Loss streak of 10 reached. Stopping...');
        engine.stop();
    }

    if (lastBet.win) {
        betSize = initialBetSize;
    } else {
        betSize *= 2;
    }
});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
});
```

Pump

Variables

- `betSize`
  Type: float
  Required: true
- `difficulty`
  Type: string
  Required: true
  Possible values: easy, medium, hard, expert
- `rounds`
  Type: integer
  Required: true
  Number range starting from 1, ending depending on the difficulty:
  - easy: 1 - 24
  - medium: 1 - 22
  - hard: 1 - 20
  - expert: 1 - 15

lastBet example

```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 6,
  "nonce": 1027,
  "active": false,
  "game": "pump",
  "win": true,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 1.6333333333333333,
  "b2bMultiplier": 1.6333333333333333,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "difficulty": "expert",
    "roundTarget": 1,
    "roundTargetMultiplier": 1.6333333333333333,
    "roundResult": 9,
    "roundResultMultiplier": 400.02
  }
  "clientSeed": "yourClientSeed",
  "serverSeed": "XXX", // Simulation Mode only
  "serverSeedHashed": "XXX" // Live Mode only
}
```

Code example

```
// Randomly change difficulty and rows after every bet

// resetStats();
// resetSeed();

function randomNumber(min, max) {
    return Math.floor(Math.random() * (max - min + 1) + min);
}

function randomRounds(difficultyIndex) {
    const maxRounds = [24, 22, 20, 15];

    return randomNumber(1, maxRounds[difficultyIndex]);
}

game = 'pump';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
difficulties = ['easy', 'medium', 'hard', 'expert'];
difficultyIndex = 0;
difficulty = difficulties[difficultyIndex];
rounds = randomRounds(difficultyIndex);

engine.onBetPlaced(async (lastBet) => {
    difficulty = difficulties[++difficultyIndex];
    rounds = randomRounds(difficultyIndex);
```