**TwoA TypeScript API Manual**

Contents

[Summary 2](#_Toc520796506)

[Adaptation Modules 3](#_Toc520796507)

[Rating Scale 4](#_Toc520796508)

[TwoA class 6](#_Toc520796509)

[Methods for adaptation. 6](#_Toc520796510)

[Methods for assessment. 8](#_Toc520796511)

[Methods for scoring. 10](#_Toc520796512)

[Methods for controlling success rate parameter. 11](#_Toc520796513)

[Methods for controlling the fuzzy selection intervals. 12](#_Toc520796514)

[Methods for the uncertainty parameter. 13](#_Toc520796515)

[Methods for K factor. 15](#_Toc520796516)

[Methods for the calibration parameters. 17](#_Toc520796517)

[Methods controlling ELO parameters. 21](#_Toc520796518)

[Methods for player data. 22](#_Toc520796519)

[Methods for scenario data. 29](#_Toc520796520)

[PlayerNode class 37](#_Toc520796521)

[ScenarioNode class 39](#_Toc520796522)

[Gameplay class 41](#_Toc520796523)

# Summary

|  |  |
| --- | --- |
| **Document information** | |
| Document version | 1.2.5 |
| Document date | 2018.07.30 |
| Document author | Enkhbold Nyamsuren  (E.Nyamsuren@gmail.com, Enkhbold.Nyamsuren@ou.nl) |

|  |  |
| --- | --- |
| **Asset author information** | |
| Name | Enkhbold Nyamsuren |
| E-mail | E.Nyamsuren@gmail.com  Enkhbold.Nyamsuren@ou.nl |
| Organization | OUNL |

|  |  |
| --- | --- |
| **Asset information** | |
| Current version | 1.2.5 |
| Date | 2018.06.18 |
| Deployment side | client-side |
| Programming language | TypeScript |
| Required libraries | Microsoft .NET 3.5 Framework or higher |
| Recommended platform | Windows OS |

# Adaptation Modules

The asset provides two different modules for adaptation and assessment. Modules differ in terms of required input to the asset from a game. In this context, input refers to player’s performance measures such as response time and accuracy. Adaptation module ID should be passed to the TwoA asset in order to indicated the module that should be used.

|  |  |
| --- | --- |
| **Adaptation Module 1** | |
| Adatation ID (string type) | “Game difficulty - Player skill” |
| Description | Assess and adapts game difficulty to player skill. Skill ratings are evaluated for individual players. Requires player accuracy and response time. Uses a modified version of the CAP algorithm. |
| Input 1 | Player’s accuracy. The value should have *double* type. The value should be either 0 or 1. The value is 1 if the player successfully completed a game scenario. The value is 0 if the player failed the game scenario. |
| Input 2 | Player’s response time. The value should have *double* type. The duration of time the player required to complete (either successfully or unsuccessfully) a game scenario. Time is measured in milliseconds. |

|  |  |
| --- | --- |
| **Adaptation Module 1** | |
| Adatation ID (string type) | “SkillDifficultyElo” |
| Description | Assess and adapts game difficulty to player skill. Skill ratings are evaluated for individual players. Requires player accuracy and response time. Uses a modified version of the CAP algorithm. |
| Input 1 | Player’s accuracy. The value should have *double* type. The value should be between 0 and 1. The value of 0 represents the worst possible player performance in a game scenario. The value of 1 represents the best possible player performance in a game scenario. |

# Rating Scale

Given a player with a skill rating *θ*, the table below shows the player’s expected success rate (column “*P*”) in a scenario with a specific difficulty rating (column “Difficulty rating”). For example, if the player’s skill rating is equal to one (*θ* = 1) then the player has 74% chance of successfully completing a scenario with a difficulty rating -0.046 (1 - 1.046).

|  |  |  |
| --- | --- | --- |
| ***P*** | ***P* in %** | **Difficulty rating** |
| 0.02 | 2% | *θ* + 3.8918 |
| 0.04 | 4% | *θ* + 3.1781 |
| 0.06 | 6% | *θ* + 2.7515 |
| 0.08 | 8% | *θ* + 2.4423 |
| 0.1 | 10% | *θ* + 2.1972 |
| 0.12 | 12% | *θ* + 1.9924 |
| 0.14 | 14% | *θ* + 1.8153 |
| 0.16 | 16% | *θ* + 1.6582 |
| 0.18 | 18% | *θ* + 1.5163 |
| 0.2 | 20% | *θ* + 1.3863 |
| 0.22 | 22% | *θ* + 1.2657 |
| 0.24 | 24% | *θ* + 1.1527 |
| 0.26 | 26% | *θ* + 1.046 |
| 0.28 | 28% | *θ* + 0.9445 |
| 0.3 | 30% | *θ* + 0.8473 |
| 0.32 | 32% | *θ* + 0.7538 |
| 0.34 | 34% | *θ* + 0.6633 |
| 0.36 | 36% | *θ* + 0.5754 |
| 0.38 | 38% | *θ* + 0.4895 |
| 0.4 | 40% | *θ* + 0.4055 |
| 0.42 | 42% | *θ* + 0.3228 |
| 0.44 | 44% | *θ* + 0.2412 |
| 0.46 | 46% | *θ* + 0.1603 |
| 0.48 | 48% | *θ* + 0.08 |
| 0.5 | 50% | *θ* + 0 |
| 0.52 | 52% | *θ* - 0.08 |
| 0.54 | 54% | *θ* - 0.1603 |
| 0.56 | 56% | *θ* - 0.2412 |
| 0.58 | 58% | *θ* - 0.3228 |
| 0.6 | 60% | *θ* - 0.4055 |
| 0.62 | 62% | *θ* - 0.4895 |
| 0.64 | 64% | *θ* - 0.5754 |
| 0.66 | 66% | *θ* - 0.6633 |
| 0.68 | 68% | *θ* - 0.7538 |
| 0.7 | 70% | *θ* - 0.8473 |
| 0.72 | 72% | *θ* - 0.9445 |
| 0.74 | 74% | *θ* - 1.046 |
| 0.76 | 76% | *θ* - 1.1527 |
| 0.78 | 78% | *θ* - 1.2657 |
| 0.8 | 80% | *θ* - 1.3863 |
| 0.82 | 82% | *θ* - 1.5163 |
| 0.84 | 84% | *θ* - 1.6582 |
| 0.86 | 86% | *θ* - 1.8153 |
| 0.88 | 88% | *θ* - 1.9924 |
| 0.9 | 90% | *θ* - 2.1972 |
| 0.92 | 92% | *θ* - 2.4423 |
| 0.94 | 94% | *θ* - 2.7515 |
| 0.96 | 96% | *θ* - 3.1781 |
| 0.98 | 98% | *θ* - 3.8918 |

# TwoA class

|  |  |
| --- | --- |
| **Class** | |
| Class name | TwoA |
| Namespace | TwoANS |
| Description | The main class of the asset. An instance of this class should be created to access asset’s API. |

|  |  |
| --- | --- |
| **Constructor** | |
| Name | constructor() |
| Description | Initializes a new instance of the TwoA class. |
|  |  |

|  |  |
| --- | --- |
| **Field name** | **Field description** |
| players: PlayerNode[] | A list of PlayerNode instances. An empty array is automatically initialized during constructor call. Each PlayerNode instance contains data of a single player. Refer to PlayerNode section for more information. |
| scenarios: ScenarioNode[] | A list of ScenarioNode instances. An empty array is automatically initialized during constructor call. Each ScenarioNode instance contains data of a single game scenario. Refer to ScenarioNode section for more information. |
| gameplays: Gameplay[] | A list of Gameplay instances. An empty array is automatically initialized during constructor call. Each Gameplay instance contains a single assessment record created at the end of UpdateRatings method. Refer to Gameplay section for more information. |

## Methods for adaptation.

|  |  |
| --- | --- |
| **Method** | |
| Name | AvailableAdapters(): string[][] |
| Description | Returns Nx3 array of string. Each row contains information about an available adaptation module. The first column contains the class name. The second column stores adaptation module ID. The third column stores a description for the adaptation module. |
| **Return type** | **Return description** |
| string[][] | 2D array with a N number of rows and 3 columns. |

|  |  |
| --- | --- |
| **Method** | |
| Name | TargetScenarioID(p\_playerNode: PlayerNode): string |
| Description | Returns ID of a game scenario with a difficulty rating that matches the skill rating of a specified player. The recommended scenario is selected from TwoA.scenarios list. |
| **Parameter name** | **Parameter description** |
| p\_playerNode: PlayerNode | PlayerNode of a player to whom the scenario difficulty should be matched. |
| **Return type** | **Return description** |
| string | ID of a game scenario of recommended difficulty. Null if any error occurred. |

|  |  |
| --- | --- |
| **Method** | |
| Name | TargetScenario(p\_playerNode: PlayerNode): ScenarioNode |
| Description | Returns an instance of ScenarioNode of a game scenario with a difficulty rating that matches the skill rating of a specified player. The recommended scenario is selected from TwoA.scenarios list. |
| **Parameter name** | **Parameter description** |
| p\_playerNode: PlayerNode | PlayerNode of a player to whom the scenario difficulty should be matched. |
| **Return type** | **Return description** |
| ScenarioNode | An instance of ScenarioNode of a game scenario of recommended difficulty. Null if any error occurred. |

|  |  |
| --- | --- |
| **Method** | |
| Name | TargetScenarioCustom(p\_playerNode: PlayerNode, p\_scenarioList: ScenarioNode[]): ScenarioNode |
| Description | Returns an instance of ScenarioNode of a game scenario with a difficulty rating that matches the skill rating of a specified player. Requires a custom list of scenarios from which a recommendation should be made. |
| **Parameter name** | **Parameter description** |
| p\_playerNode: PlayerNode | PlayerNode of a player to whom the scenario difficulty should be matched. |
| p\_scenarioList: ScenarioNode[] | A list of ScenarioNode instances from which a scenario will be selected and matched to player’s skill level. |
| **Return type** | **Return description** |
| ScenarioNode | An instance of ScenarioNode of a game scenario of recommended difficulty. Null if any error occurred. |

|  |  |
| --- | --- |
| **Method** | |
| Name | TargetDifficultyRatingCustom(p\_adaptID: string, p\_playerRating: number): number |
| Description | Returns a recommended difficulty rating for a specified player skill rating. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (refer to “Adaptation modules”). |
| p\_playerRating: number | Player’s skill rating. |
| **Return type** | **Return description** |
| number | Difficulty rating. |

|  |  |
| --- | --- |
| **Method** | |
| Name | TargetDifficultyRating(p\_playerNode: PlayerNode): number |
| Description | Returns a recommended difficulty rating for a specified player. |
| **Parameter name** | **Parameter description** |
| p\_playerNode: PlayerNode | PlayerNode of a player to whom the scenario difficulty should be matched. |
| **Return type** | **Return description** |
| number | Difficulty rating. |

## Methods for assessment.

|  |  |
| --- | --- |
| **Method** | |
| Name | UpdateRatings(p\_playerNode: PlayerNode, p\_scenarioNode: ScenarioNode, p\_rt: number, p\_correctAnswer: number, p\_updateScenarioRating: boolean, p\_customKfct: number): boolean |
| Description | Reassesses and updates player skill rating and, optionally, scenario difficulty rating based on player’s performance in a specified scenario. |
| **Parameter name** | **Parameter description** |
| p\_playerNode: PlayerNode | PlayerNode of a player to assess. |
| p\_scenarioNode: ScenarioNode | ScenarioNode a scenario in which player’s performance was measure. |
| p\_rt: number | Player’s response time measured in milliseconds (see “Adaptation Modules”). If adaptation is based on accuracy only then this parameter will be automatically ignored. Should be higher than 0. |
| p\_correctAnswer: number | Player’s accuracy (see “Adaptation Modules”). Depending on the adaptation module should be either binary (0 or 1) or a value between 0 and 1. Assessment is not performed if accuracy value does not match module’s requirements. |
| p\_updateScenarioRating: boolean | If true scenario’s difficulty rating will be reassessed and updated. |
| p\_customKfct: number | A custom K factor to control the scale of changes in player’s and scenario’s ratings. Requires a positive value. A higher value results in a bigger change in the rating. If value is 0 then TwoA uses a dynamically estimated K factor. Use this parameter with care since it can drastically influence TwoA’s performance of adaptation and assessment. Consult the table in section “Rating Scale” to decide on the K factor appropriate for you. |
| **Return type** | **Return description** |
| boolean | True if ratings were reassessed and updated successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | CreateNewRecord(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_scenarioID: string, p\_rt: number, p\_accuracy: number, p\_playerRating: number, p\_scenarioRating: number, p\_timestamp: string): void |
| Description | Records results of player assessment by creating a new instance of Gameplay class. The instance is stored in TwoA.gameplays list. The player and scenario data is retrieved from the TwoA.players and TwoA.scenarios lists respectively. This method is automatically called by UpdateRatings methods. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game from which a scenario was selected. |
| p\_playerID: string | ID of a player that was assessed. |
| p\_scenarioID: string | ID of a scenario in which player’s performance was measured. |
| p\_rt: number | Player’s response time in milliseconds (see “Adaptation Modules”). |
| p\_accuracy: number | Player’s accuracy (see “Adaptation Modules”). Depending on the adaptation module should be either binary (0 or 1) or a value between 0 and 1. |
| p\_playerRating: number | Player’s skill rating after reassessment. |
| p\_scenarioRating: number | Scenario’s difficulty rating after reassessment. |
| p\_timestamp: string | Date and time of reassessment. |

## Methods for scoring.

|  |  |
| --- | --- |
| **Method** | |
| Name | CalculateScore(p\_correctAnswer: number, p\_responseTime: number, p\_itemMaxDuration: number): number |
| Description | Transforms player’s accuracy and response time into a single score measured in the range (-1, 1). |
| **Parameter name** | **Parameter description** |
| p\_correctAnswer: number | Player’s accuracy that is either 0 or 1. 1 is for success, and 0 is for fails. |
| p\_responseTime: number | Player’s response time in milliseconds. |
| p\_itemMaxDuration: number | Max amount of time a player is allowed to spend to complete a game scenario. Measured in milliseconds. |
| **Return type** | **Return description** |
| number | A score between -1 and 1. |

|  |  |
| --- | --- |
| **Method** | |
| Name | CalculateExpectedScore(p\_adaptID: string, p\_playerRating: number, p\_scenarioRating: number, p\_itemMaxDuration: number): number |
| Description | Calculates player's expected score based on player's skill rating and scenarios difficulty rating. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of the adaptation module to use. |
| p\_playerRating: number | Player’s skill rating. |
| p\_scenarioRating: number | Scenario’s difficulty rating. |
| p\_itemMaxDuration: number | Max allowed time in millisecond given to player to solve the problem. |
| **Return type** | **Return description** |
| number | A score between -1 and 1. |

## Methods for controlling success rate parameter.

When recommending scenarios, TwoA tries to ensure that a player can maintain an average success rate *P* of successfully completing the scenarios. For example, if *P* = 0.75 then the player is ideally expected to successfully complete 75% of all scenarios recommended by the TwoA asset. In a more realistic case, player’s actual success rate will not be exactly 75% but close to it (little bit more or little bit less). More specifically, player’s real success rate will follow a normal distribution with mean at 0.75.

The game developer can change the success rate parameters to suit needs of specific games or player audience. Four parameters are needed to set the success rate. The first two are the mean and standard deviation defining the normal distribution. There are also two hard boundaries within which player’s real success rate is expected to lie.

The default success rate is defined by a normal distribution N(M=0.75, SD=0.1) with mean at 0.75 and a standard deviation of 0.1. Hard boundaries are 0.5 and 0.99. This means there is 95% chance that player actual success rate will be between 55% (M – 2\*SD) and 95% (M + 2\*SD), but it will never drop below 50% and never go above 99%.

Similarly, a game developer can set success rate to follow a normal distribution N(M=0.5, SD=0.1) with hard boundaries at 0.25 and 0.75. This means there is 95% chance that player actual success rate will be between 30% (M – 2\*SD) and 70% (M + 2\*SD), but it will never drop below 25% and never go above 75%.

|  |  |
| --- | --- |
| **Method** | |
| Name | GetTargetDistribution(p\_adaptID: string): number[] |
| Description | Returns the four parameters defining the target success rate that is used to select a scenario of a recommended difficulty. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number[] | An array of four double values: distribution mean, distribution standard deviation, lower hard boundary, and upper hard boundary. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetTargetDistribution(p\_adaptID: string, p\_mean: number, p\_sd: number, p\_lowerLimit: number, p\_upperLimit: number): void |
| Description | Sets the parameters for the target success rate that is used to select a scenario of a recommended difficulty. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_mean: number | Mean of a normal distribution. Any value between 0 and 1 (exclusive). |
| p\_sd: number | Standard deviation of a normal distribution. Any value between 0 and 1 (exclusive). |
| p\_lowerLimit: number | Lower hard boundary. Any value between 0 and 1 (inclusive). Should be less than standard distribution mean. |
| p\_upperLimit: number | Upper hard boundary. Any value between 0 and 1 (inclusive). Should be higher than standard distribution mean. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultTargetDistribution(p\_adaptID: string): void |
| Description | Sets the parameters for the target success rate to its default values that is used to select a scenario of a recommended difficulty. The default values are 0.75 for distribution mean, 0.1 for standard deviation, 0.5 for lower boundary, and 1 for upper boundary. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

## Methods for controlling the fuzzy selection intervals.

|  |  |
| --- | --- |
| **Method** | |
| Name | GetFiSDMultiplier(p\_adaptID: string): number |
| Description | Returns the multiplier for the standard deviation used for calculating the support fuzzy interval. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | Multiplier value, or 0 if the adapter is not found. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetFiSDMultiplier(p\_adaptID: string, p\_multiplier: number): void |
| Description | Sets a value for the multiplier for the standard deviation used for calculating the support fuzzy interval. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_multiplier: number | The value of the multiplier. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultFiSDMultiplier(p\_adaptID: string): void |
| Description | Sets to its default value the multiplier for the standard deviation used for calculating the support fuzzy interval. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

## Methods for the uncertainty parameter.

|  |  |
| --- | --- |
| **Method** | |
| Name | GetMaxDelay(p\_adaptID: string): number |
| Description | Gets value for the max number of days after which player's or item's uncertainty reaches the maximum of 1. This a continuous number of days during which the player has not played any scenario, or the scenario was not played by any player. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | The number of days as double value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetMaxDelay(p\_adaptID: string, p\_maxDelay: number): void |
| Description | Sets a value for the max number of days after which player's or item's uncertainty reaches the maximum of 1. This a continuous number of days during which the player has not played any scenario, or the scenario was not played by any player. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_maxDelay: number | The value in the number of days. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultMaxDelay(p\_adaptID: string): void |
| Description | Sets to its default value the max number of days after which player's or item's uncertainty reaches the maximum of 1. This a continuous number of days during which the player has not played any scenario, or the scenario was not played by any player. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetMaxPlay(p\_adaptID: string): number |
| Description | Gets value for the max number of gameplays after which player’s or scenario’s rating uncertainty reaches 0. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | The number of gameplays as double value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetMaxPlay(p\_adaptID: string, p\_maxPlay: number): void |
| Description | Sets a value for the max number of gameplays after which player’s or scenario’s rating uncertainty reaches 0. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_maxPlay: number | The value in the number of gameplays. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultMaxPlay(p\_adaptID: string): void |
| Description | Sets to its default value the max number of gameplays after which player’s or scenario’s rating uncertainty reaches 0. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

## Methods for K factor.

|  |  |
| --- | --- |
| **Method** | |
| Name | GetKConst(p\_adaptID: string): number |
| Description | Gets the min value for the K factor that is used when there are no uncertainties in player’s and scenario’s ratings. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | K factor value as double. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetKConst(p\_adaptID: string, p\_kConst: number): void |
| Description | Sets the min value for the K factor that is used when there are no uncertainties in player’s and scenario’s ratings. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_kConst: number | The min value for the K factor. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultKConst(p\_adaptID: string): void |
| Description | Sets the min value for the K factor to its default value. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetKUp(p\_adaptID: string): number |
| Description | Gets the step value by which the K factor should increase. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | K factor step value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetKUp(p\_adaptID: string, p\_kUp: number): void |
| Description | Sets the step value by which the K factor should increase. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_kUp: number | The step value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultKUp(p\_adaptID: string): void |
| Description | Sets to its default value the step size by which the K factor should increase. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetKDown(p\_adaptID: string): number |
| Description | Gets the step value by which the K factor should decrease. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | K factor step value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetKDown(p\_adaptID: string, p\_kDown: number): void |
| Description | Sets the step value by which the K factor should decrease. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_kDown: number | The step value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultKDown(p\_adaptID: string): void |
| Description | Sets to its default value the step size by which the K factor should decrease. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

## Methods for the calibration parameters.

|  |  |
| --- | --- |
| **Method** | |
| Name | GetPlayerCalLength(p\_adaptID: string): number |
| Description | Returns the length of a player’s calibration phase. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | Number of gameplays. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetPlayerCalLength(p\_adaptID: string, p\_calLength: number): void |
| Description | Sets the length of a player’s calibration phase. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_calLength: number | The length in the number of gameplays. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultPlayerCalLength(p\_adaptID: string): void |
| Description | Sets to its default value the length of a player’s calibration phase. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioCalLength(p\_adaptID: string): number |
| Description | Returns the length of a scenario’s calibration phase. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | Number of gameplays. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioCalLength(p\_adaptID: string, p\_calLength: number): void |
| Description | Sets the length of a scenario’s calibration phase. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_calLength: number | The length in the number of gameplays. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultScenarioCalLength(p\_adaptID: string): void |
| Description | Sets to its default value the length of a scenario’s calibration phase. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetCalLength(p\_adaptID: string, p\_calLength: number): void |
| Description | Sets the scenario and player calibration lengths to the same value. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_calLength: number | The length in the number of gameplays. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultCalLength(p\_adaptID: string): void |
| Description | Sets scenario and player calibration lengths to its default values. The length is measured in number of gameplays. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetPlayerCalK(p\_adaptID: string): number |
| Description | Returns the custom K factor used during player’s calibration. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | K factor value as double. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetPlayerCalK(p\_adaptID: string, p\_calK: number): void |
| Description | Sets the custom K factor used during player’s calibration. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_calK: number | K factor value as double. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultPlayerCalK(p\_adaptID: string): void |
| Description | Sets to its default value the custom K factor used during player’s calibration. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioCalK(p\_adaptID: string): number |
| Description | Returns the custom K factor used during scenario’s calibration. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | K factor value as double. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioCalK(p\_adaptID: string, p\_calK: number): void |
| Description | Sets the custom K factor used during scenario’s calibration. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_calK: number | K factor value as double. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultScenarioCalK(p\_adaptID: string): void |
| Description | Sets to its default value the custom K factor used during scenario’s calibration. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetCalK(p\_adaptID: string, p\_calK: number): void |
| Description | Sets the custom K factor used during both player’s and scenario’s calibrations. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_calK: number | K factor value as double. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultCalK(p\_adaptID: string): void |
| Description | Sets to its default values the custom K factors used during player’s and scenario’s calibrations. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

## Methods controlling ELO parameters.

|  |  |
| --- | --- |
| **Method** | |
| Name | GetExpectScoreMagnifier(p\_adaptID: string): number |
| Description | Returns the value of the magnifier for the expected score compared to the opponent’s score. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | Magnifier value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetExpectScoreMagnifier(p\_adaptID: string, p\_expectScoreMagnifier: number): void |
| Description | Sets the value of the magnifier for the expected score compared to the opponent’s score. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_expectScoreMagnifier: number | Magnifier value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultExpectScoreMagnifier(p\_adaptID: string): void |
| Description | Sets to its default value the magnifier for the expected score compared to the opponent’s score. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetMagnifierStepSize(p\_adaptID: string): number |
| Description | Returns the value of the magnifier step size. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| **Return type** | **Return description** |
| number | Magnifier step size value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetMagnifierStepSize(p\_adaptID: string, p\_magnifierStepSize: number): void |
| Description | Sets the value of the magnifier step size. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_magnifierStepSize: number | Magnifier step size value. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetDefaultMagnifierStepSize(p\_adaptID: string): void |
| Description | Sets the magnifier step size to its default value. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |

## Methods for player data.

|  |  |
| --- | --- |
| **Method** | |
| Name | AddPlayer(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_rating: number, p\_playCount: number, p\_kFactor: number, p\_uncertainty: number, p\_lastPlayed: string): PlayerNode |
| Description | Creates a new instance of PlayerNode and adds it to the TwoA.players list. Requires custom parameter values. Ensures that all player parameters have valid values before creating the instance. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game to which the player instance belongs. |
| p\_playerID: string | ID of a player. This ID is not allowed to duplicate for the same combination of the gameID and adaptID already present in the TwoA.players list. |
| p\_rating: number | Player’s skill rating. |
| p\_playCount: number | The number of past gameplays that were used to assess player’s skill rating. Should be a non-0 value. |
| p\_kFactor: number | K factor. Should be higher than 0. |
| p\_uncertainty: number | Uncertainty in player’s rating. Should be a value between 0 and 1 (inclusive). |
| p\_lastPlayed: string | The datetime of the last gameplay that was used to assess player’s skill rating. |
| **Return type** | **Return description** |
| PlayerNode | Returns the newly created instance of the PlayerNode class. |

|  |  |
| --- | --- |
| **Method** | |
| Name | AddPlayerDefault(p\_adaptID: string, p\_gameID: string, p\_playerID: string): PlayerNode |
| Description | Creates a new instance of PlayerNode and adds it to the TwoA.players list. Assigns default values to all player parameters. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game to which the player instance belongs. |
| p\_playerID: string | ID of a player. This ID is not allowed to duplicate for the same combination of the gameID and adaptID already present in the TwoA.players list. |
| **Return type** | **Return description** |
| PlayerNode | Returns the newly created instance of the PlayerNode class. |

|  |  |
| --- | --- |
| **Method** | |
| Name | AddPlayerNode(p\_playerNode: PlayerNode): boolean |
| Description | Adds the instance of PlayerNode to the TwoA.players list. Ensures that all player parameters have valid values before adding the instance. |
| **Parameter name** | **Parameter description** |
| p\_playerNode: PlayerNode | PlayerNode instance with new player data. |
| **Return type** | **Return description** |
| boolean | True if a new instance was successfully added, and False otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | RemovePlayer(p\_adaptID: string, p\_gameID: string, p\_playerID: string): boolean |
| Description | Removes the matching instance of PlayerNode from the TwoA.players list. The instance is matched by the ID combination. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| **Return type** | **Return description** |
| boolean | True if the instance was removed successfully, and False otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | RemovePlayerNode(p\_playerNode: PlayerNode): boolean |
| Description | Removes the instance of PlayerNode from the TwoA.players list. |
| **Parameter name** | **Parameter description** |
| p\_playerNode: PlayerNode | The instance to remove. |
| **Return type** | **Return description** |
| boolean | True if the instance was removed successfully, and False otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | Player(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_showWarning: boolean = true): PlayerNode |
| Description | Returns the matching instance of PlayerNode from the TwoA.players list. The instance is matched by the ID combination. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| p\_showWarning: boolean | If true and if no matching player can be found then prints warning in the log. |
| **Return type** | **Return description** |
| PlayerNode | The matching instance. Null if no match is found or error occurred. |

|  |  |
| --- | --- |
| **Method** | |
| Name | AllPlayers(p\_adaptID: string, p\_gameID: string): PlayerNode[] |
| Description | Returns the all matching instances of PlayerNode from the TwoA.players list. The instances are matched by the ID combination. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| **Return type** | **Return description** |
| PlayerNode[] | The list of matching instances. Null if no match is found or error occurred. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetPlayerRating(p\_adaptID: string, p\_gameID: string, p\_playerID: string): number |
| Description | Returns the skill rating for the specified player. Throws ReferenceError if PlayerNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| **Return type** | **Return description** |
| number | Skill rating. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetPlayerPlayCount(p\_adaptID: string, p\_gameID: string, p\_playerID: string): number |
| Description | Returns the play count for the specified player. Throws ReferenceError if PlayerNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| **Return type** | **Return description** |
| number | Play count. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetPlayerKFactor(p\_adaptID: string, p\_gameID: string, p\_playerID: string): number |
| Description | Returns the K factor for the specified player. Throws ReferenceError if PlayerNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| **Return type** | **Return description** |
| number | K factor. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetPlayerUncertainty(p\_adaptID: string, p\_gameID: string, p\_playerID: string): number |
| Description | Returns the rating uncertainty for the specified player. Throws ReferenceError if PlayerNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| **Return type** | **Return description** |
| number | Rating uncertainty. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetPlayerLastPlayed(p\_adaptID: string, p\_gameID: string, p\_playerID: string): string |
| Description | Returns a string representation of datetime indicating the last timestamp the player’s skill rating was (re)assessed. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| **Return type** | **Return description** |
| string | String representation of datetime. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetPlayerRating(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_rating: number): boolean |
| Description | Sets the skill rating for the specified player. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| p\_rating: number | New skill rating value. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetPlayerPlayCount(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_playCount: number): boolean |
| Description | Sets the play count for the specified player. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| p\_playCount: number | New play count. Positive Integer value. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetPlayerKFactor(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_kFactor: number): boolean |
| Description | Sets the K factor for the specified player. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| p\_kFactor: number | New K factor. Positive non-0 value. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetPlayerUncertainty(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_uncertainty: number): boolean |
| Description | Sets the rating uncertainty for the specified player. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| p\_uncertainty: number | New uncertainty. Value between 0 and 1 (inclusive). |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetPlayerLastPlayed(p\_adaptID: string, p\_gameID: string, p\_playerID: string, p\_lastPlayed: string): boolean |
| Description | Sets the datetime indicating the last timestamp the player’s skill rating was (re)assessed. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_playerID: string | ID of a player. |
| p\_lastPlayed: string | String representation of datetime of the last (re)assessment. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

## Methods for scenario data.

|  |  |
| --- | --- |
| **Method** | |
| Name | AddScenario(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_rating: number, p\_playCount: number, p\_kFactor: number, p\_uncertainty: number, p\_lastPlayed: string, p\_timeLimit: number): ScenarioNode |
| Description | Creates a new instance of ScenarioNode and adds it to the TwoA.scenarios list. Requires custom parameter values. Ensures that all scenario parameters have valid values before creating the instance. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game to which the scenario instance belongs. |
| p\_scenarioID: string | ID of a scenario. This ID is not allowed to duplicate for the same combination of the gameID and adaptID already present in the TwoA.scenarios list. |
| p\_rating: number | Scenario’s skill rating. |
| p\_playCount: number | The number of past gameplays that were used to assess scenario’s difficulty rating. Should be a non-0 value. |
| p\_kFactor: number | K factor. Should be higher than 0. |
| p\_uncertainty: number | Uncertainty in scenario’s rating. Should be a value between 0 and 1 (inclusive). |
| p\_lastPlayed: string | The datetime of the last gameplay that was used to assess scenario’s difficulty rating. |
| p\_timeLimit: number | Time limit within which a player should complete the scenario. Measured in milliseconds. |
| **Return type** | **Return description** |
| ScenarioNode | Returns the newly created instance of the ScenarioNode class. |

|  |  |
| --- | --- |
| **Method** | |
| Name | AddScenarioDefault(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): ScenarioNode |
| Description | Creates a new instance of ScenarioNode and adds it to the TwoA.scenarios list. Assigns default values to all scenario parameters. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game to which the scenario instance belongs. |
| p\_scenarioID: string | ID of a scenario. This ID is not allowed to duplicate for the same combination of the gameID and adaptID already present in the TwoA.scenarios list. |
| **Return type** | **Return description** |
| ScenarioNode | Returns the newly created instance of the ScenarioNode class. |

|  |  |
| --- | --- |
| **Method** | |
| Name | AddScenarioNode(p\_scenarioNode: ScenarioNode): boolean |
| Description | Adds the instance of ScenarioNode to the TwoA.scenarios list. Ensures that all scenarios parameters have valid values before adding the instance. |
| **Parameter name** | **Parameter description** |
| p\_scenarioNode: ScenarioNode | ScenarioNode instance with data for the new scenario. |
| **Return type** | **Return description** |
| boolean | True if a new instance was successfully created, and False otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | RemoveScenario(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): boolean |
| Description | Removes the matching instance of ScenarioNode from the TwoA.scenarios list. The instance is matched by the ID combination. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| **Return type** | **Return description** |
| boolean | True if the instance was removed successfully, and False otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | RemoveScenarioNode(p\_scenarioNode: ScenarioNode): boolean |
| Description | Removes the instance of ScenarioNode from the TwoA.scenarios list. |
| **Parameter name** | **Parameter description** |
| p\_scenarioNode: ScenarioNode | The instance to remove. |
| **Return type** | **Return description** |
| boolean | True if the instance was removed successfully, and False otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | Scenario(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_showWarning: boolean = true): ScenarioNode |
| Description | Returns the matching instance of ScenarioNode from the TwoA.scenarios list. The instance is matched by the ID combination. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| p\_showWarning: boolean | If true and if no matching scenario can be found then prints warning in the log. |
| **Return type** | **Return description** |
| ScenarioNode | The matching instance. Null if no match is found or error occurred. |

|  |  |
| --- | --- |
| **Method** | |
| Name | AllScenarios(p\_adaptID: string, p\_gameID: string): ScenarioNode[] |
| Description | Returns the all matching instances of ScenarioNode from the TwoA.scenarios list. The instances are matched by the ID combination. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| **Return type** | **Return description** |
| ScenarioNode[] | The list of matching instances. Null if no match is found or error occurred. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioRating(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): number |
| Description | Returns the difficulty rating for the specified scenario. Throws ReferenceError if ScenarioNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| **Return type** | **Return description** |
| number | Difficulty rating. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioPlayCount(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): number |
| Description | Returns the play count for the specified scenario. Throws ReferenceError if ScenarioNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| **Return type** | **Return description** |
| number | Play count. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioKFactor(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): number |
| Description | Returns the K factor for the specified scenario. Throws ReferenceError if ScenarioNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| **Return type** | **Return description** |
| number | K factor. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioUncertainty(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): number |
| Description | Returns the rating uncertainty for the specified scenario. Throws ReferenceError if ScenarioNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| **Return type** | **Return description** |
| number | Rating uncertainty. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioLastPlayed(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): string |
| Description | Returns a string representing datetime indicating the last timestamp the scenario’s difficulty rating was (re)assessed. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| **Return type** | **Return description** |
| string | String representing datetime. |

|  |  |
| --- | --- |
| **Method** | |
| Name | GetScenarioTimeLimit(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string): number |
| Description | Returns the time limit for the specified scenario. Throws ReferenceError if ScenarioNode instance is not found. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| **Return type** | **Return description** |
| number | Time limit. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioRating(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_rating: number): boolean |
| Description | Sets the difficulty rating for the specified scenario. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| p\_rating: number | New difficulty rating value. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioPlayCount(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_playCount: number): boolean |
| Description | Sets the play count for the specified scenario. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| p\_playCount: number | New play count. Positive Integer value. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioKFactor(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_kFactor: number): boolean |
| Description | Sets the K factor for the specified scenario. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| p\_kFactor: number | New K factor. Positive non-0 value. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioUncertainty(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_uncertainty: number): boolean |
| Description | Sets the rating uncertainty for the specified scenario. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| p\_uncertainty: number | New uncertainty. Value between 0 and 1 (inclusive). |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioLastPlayed(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_lastPlayed: string): boolean |
| Description | Sets the datetime indicating the last timestamp the scenario’s difficulty rating was (re)assessed. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| p\_lastPlayed: string | Datetime of the last (re)assessment. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

|  |  |
| --- | --- |
| **Method** | |
| Name | SetScenarioTimeLimit(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string, p\_timeLimit: number): boolean |
| Description | Sets the time limit for the specified scenario. |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game. |
| p\_scenarioID: string | ID of a scenario. |
| p\_timeLimit: number | New time limit. Positive non-0 value. |
| **Return type** | **Return description** |
| boolean | True if parameter value was set successfully, and false otherwise. |

# PlayerNode class

|  |  |
| --- | --- |
| **Class** | |
| Class name | PlayerNode |
| Namespace | TwoANS |
| Description | Stores adaptation and assessment data for a player. An instance of this class is stored in ‘public players: PlayerNode[]’ of the TwoA class. |

|  |  |
| --- | --- |
| **Constructor** | |
| Name | constructor(p\_adaptID: string, p\_gameID: string, p\_playerID: string) |
| Description | Initializes a new instance of the PlayerNode class with default parameter values. |
|  |  |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game to which the player instance belongs. |
| p\_playerID: string | ID of a player. |

|  |  |
| --- | --- |
| **Property name** | **Property description** |
| AdaptationID: string | Gets or sets ID of an adaptation module to be used (see “Adaptation Modules”). |
| GameID: string | Gets or sets ID of a game to which the player instance belongs. |
| PlayerID: string | Gets or sets ID of a player. |
| Rating: number | Gets or sets player’s skill rating. |
| PlayCount: number | Gets or sets the number of past gameplays that were used to assess player’s skill rating. Should be an integer that is 0 or higher. |
| KFactor: number | Gets or sets the K factor. Should be a double that is higher than 0. |
| Uncertainty: number | Gets or sets the uncertainty in player’s rating. Should be a value between 0 and 1 (inclusive at both ends). |
| LastPlayed: string | Gets or sets the datetime of the last gameplay that was used to assess player’s skill rating. |

|  |  |
| --- | --- |
| **Method** | |
| Name | PlayerNode ShallowClone() |
| Description | Creates and returns a shallow clone of the instance. |
| **Return type** | **Return description** |
| PlayerNode | New instance of PlayerNode. |

# ScenarioNode class

|  |  |
| --- | --- |
| **Class** | |
| Class name | ScenarioNode |
| Namespace | TwoANS |
| Description | Stores adaptation and assessment data for a scenario. An instance of this class is stored in ‘public scenarios: ScenarioNode[]’ of the TwoA class. |

|  |  |
| --- | --- |
| **Constructor** | |
| Name | constructor(p\_adaptID: string, p\_gameID: string, p\_scenarioID: string) |
| Description | Initializes a new instance of the ScenarioNode class with default parameter values. |
|  |  |
| **Parameter name** | **Parameter description** |
| p\_adaptID: string | ID of an adaptation module to be used (see “Adaptation Modules”). |
| p\_gameID: string | ID of a game to which the scenario instance belongs. |
| p\_scenarioID: string | ID of a scenario. |

|  |  |
| --- | --- |
| **Property name** | **Property description** |
| AdaptationID: string | Gets or sets ID of an adaptation module to be used (see “Adaptation Modules”). |
| GameID: string | Gets or sets ID of a game to which the player instance belongs. |
| ScenarioID: string | Gets or sets ID of a scenario. |
| Rating: number | Gets or sets scenario’s difficulty rating. |
| PlayCount: number | Gets or sets the number of past gameplays that were used to assess scenario’s difficulty rating. Should be an integer that is 0 or higher. |
| KFactor: number | Gets or sets the K factor. Should be double value higher than 0. |
| Uncertainty: number | Gets or sets the uncertainty in scenario’s rating. Should be a double value between 0 and 1 (inclusive at both ends). |
| LastPlayed: string | Gets or sets the datetime of the last gameplay that was used to assess scenario’s difficulty rating. |
| TimeLimit: number | Gets or sets the time limit within which a player should complete the scenario. Measured in milliseconds. Should be an integer higher than 0. |

|  |  |
| --- | --- |
| **Method** | |
| Name | ShallowClone(): ScenarioNode |
| Description | Creates and returns a shallow clone of the instance. |
| **Return type** | **Return description** |
| ScenarioNode | New instance of ScenarioNode. |

# Gameplay class

|  |  |
| --- | --- |
| **Class** | |
| Class name | Gameplay |
| Namespace | TwoANS |
| Description | Stores results of a player assessment. An instance of this class is stored in ‘public gameplays: Gameplay[]’ of the TwoA class. |

|  |  |
| --- | --- |
| **Constructor** | |
| Name | constructor() |
| Description | Initializes a new instance of the Gameplay class. |

|  |  |
| --- | --- |
| **Property name** | **Property description** |
| AdaptationID: string | Gets or sets ID of an adaptation module to be used (see “Adaptation Modules”). |
| GameID: string | Gets or sets ID of a game to which the player instance belongs. |
| PlayerID: string | Gets or sets ID of a player that was assessed. |
| ScenarioID: string | Gets or sets ID of a scenario in which player’s performance was measured. |
| Timestamp: string | Gets or sets the datetime of assessment in the format of ‘yyyy-MM-ddThh:mm:ss’. |
| RT: number | Gets or sets the player’s response time. Measured in milliseconds. |
| Accuracy: number | Gets or sets the player’s accuracy. A value between 0 and 1 inclusive. |
| PlayerRating: number | Gets or sets the player’s skill rating after assessment. |
| ScenarioRating: number | Gets or sets the scenario’s difficulty rating after assessment. |