**Lead Caravan: Game Overview**

***Objective:***

* Amass and sell resources to earn the most points at the end of the game.

***Game structure:***

* **Groups:**
  + Players are divided into groups, and work collaboratively to win points for their team. All players in a group share the same collection of resources and same amount of cash on hand.
  + Players each have their own hand-held devices, and each group has its own base. This base shows up-to-date stock prices, the current market value, and the cash on hand for each group. Stock prices are unique to each group.
* **Stocks:**
  + Each group starts out with a collection of stocks or resources. Each year, resources are replenished. The amount of resources replenished per year is listed at the group’s base.
  + There are two classes of resources: expensive resources (like ruby and diamond) and cheap resources (like wood and stone).
  + Resources fluctuate in price.
* **Points:**
  + Points are awarded at the end of each year (one year in game time takes 6 minutes) based on cash on hand at the end of the year.
  + 1st place is awarded 12 points, 2nd place is awarded 9 points, 3rd place is awarded 4 points, and 4th place is awarded 1 point.
  + Cash on hand is reset to 0 at the end of each year after points are allocated.

***Player actions:***

* **Trades:**
  + Players can offer trades to others, specifying which resources they would to exchange.
    - As stock prices are unique to each team, it is possible to find trades which benefit both parties.
  + When you request a trade, you must input your current location, called a zone. This is a 3-digit number that will be found in the zone.
  + Players can accept or reject trades that are offered to them. When a trade is accepted, resources are immediately exchanged between groups.
* **Cashing Out:** 
  + At their base, players can cash out resources. Each resource can only be cashed out once per year. When a resource is cashed out, the group will receive the current stock price for that resource times the amount cashed in.

**Lead Caravan: Telemetry Key**

**Variables stored for each event**

|  |  |
| --- | --- |
| *Variable* | *Description and Possible Values* |
| timestamp | All events are given a timestamp. This is stored the epoch time in milliseconds. |
| key | This identifies the type of event that has occurred. Possible values include:   * “NewGameStart”: a new game has started; this event will only be recorded once at the start of a game * “StockPriceChange”: stock price just changed * “GameYearIncrease”: a year has ended, and the year number will increase * “TradeRequestSent” * “TradeRequestResponded” * “CashingOutResource”: a group has just cashed out a resource * “AnnualPointAward”: occurs at the end of the year; points are allocated to the different groups based on who has the most cash on hand * “EndYearRankStatus”: occurs at end of year after points have been awarded; gives you an update on the ranking of the different groups |
| description | Includes additional information about the event that occurred. Different keys will have different values for this variable. |
| gameCode | Each game has a unique, 4-digit code. This code is logged for all events, and can be used to filter data from different games. |
| year | Year in the game during which this event occurred. One year in the game takes 6 minutes to play. |
| user | When an event is initiated by a user (like a trade request or response), their user id is stored in the event log. |
| groupName, group | Each player belongs to a group. Each group has a name (like red group) which is stored as “groupName” and a group id number which is stored as “group”. |
| item, itemNo | Each resource is identified by a name (like “ruby” or “diamond”). These resources are also identified by codes, or itemNo. When an event involving a resource occurs, information about the resource involved is recorded. |

**Variables specific to ‘StockPriceChange’ events:**

|  |  |  |
| --- | --- | --- |
| *Variable* | *Description and Possible Values* | *When logged?* |
| description | This tells you why a price change has occurred for a given stock.   * “NewYearUpdates”: Each year, all of the stocks are updated in price. Some stocks will increase in value, while others will decrease. * “RegularUpdate”: Stocks are routinely checked to make sure that prices are correct. * “TradeCausedUpdate”: When a trade is made, the price of the stocks involved for the groups who made the trade changes. As a group’s supply of a resource increases, the price of the resource will decrease (and vice versa). * "This is the change of this group's total market value": Changes in total market value are also stored under this key. |  |
| price | New price for the stock.  If itemNo=555 and description="This is the change of this group's total market value", then this is the total market value of the group. | Stored when description is “NewYearUpdates” or  “RegularUpdate” or "This is the change of this group's total market value" |
| oldPrice | Price of the item for the listed groups before trade. | Stored when description is “TradeCausedUpdate” |
| newPrice | Price of the item for the listed group after the trade. | Stored when description is “TradeCausedUpdate” |
| context | All the information about the trade that caused this update. | Stored when description is “TradeCausedUpdate” |
| All the info about the stocks before/after the update. Nested values include “context.stockBeforeUpdate” and “context.stockAfterUpdate” | Stored when description is “RegularUpdate” |

**Variables specific to ‘TradeRequestSent’ and ‘TradeRequestResponded’**

|  |  |  |
| --- | --- | --- |
| Variable | Description and Possible Values |  |
| zone | This tells you where in the room the user was they made or accepted a trade request. A mapping of zone code to the layout of the room can be found in the dropbox folder (zonemapping.csv). |  |
| requester, requesterName, requesterGroup | Id, name, and group id of player who made the request |  |
| recipient, recipientName, recipientGroup | Id, name, and group id of player who received/responded to the request |  |
| reqResNo, reqRes, reqAmt | Number, name, and amount of resource that the requester has asked for |  |
| recvResNo, recvRes, recvAmt | Number, name, and amount of resource that the requester has offered to the recipient |  |
| reqLogContents | All the information about the initial request. |  |
| response | Response = 1 means trade was accepted  Response = 0 means trade was not responded to  Response = -1 means trade was not accepted  Response = -2 means trade was cancelled by the sender | Stored when key = “Trade  Request Responded” |
| reqResRequester | Information about the requested resource as seen by the person making the request.  This includes a variable called “amount” which is the amount of the resource that this group has on hand (not the amount being exchanged in the trade). This also includes the price of the variable for the requesting group. |  |
| recvResRequester | Information about resource to be given away as seen by the person making the request. |  |
| reqResRecipient | Information about the requested resource as seen by the person receiving the request. |  |
| recvResRecipient | Information about resource to be given away as seen by the person receiving the request. |  |

Example: Vishesh (requester) asks Lauren (recipient of the trade request for 10 ruby (reqRes = ruby, reqAmt = 10) in exchange for 8 platinum (recvRes = platinum, recvAmt = 8).

* To find out how much rubies are worth to Vishesh, I would look under reqResRequester (since ruby is the requested resource, reqRes, and Vishesh is the requester).
* To find out how much platinum is worth to Lauren, I would look under recvResRecipient (since platinum is the received resource, recvRes, and Lauren in the recipient of the trade request).

**‘CashingOutResources’:** Whenever a group cashes out a resource, data is logged under this key

|  |  |
| --- | --- |
| *Variable* | *Description and Possible Values* |
| resource | Item number of resource being cashed out |
| resourceName | Name of resource being cashed out |
| amount | Amount of resource being cashed out |

**‘AnnualPointAward’**: At the end of each year, points are awarded to the 4 groups with the most cash. A separate entry is recorded for each group who earns points.

|  |  |
| --- | --- |
| *Variable* | *Description and Possible Values* |
| cash | Amount of cash on hand at the end of the year (before value is reset). |
| pointsAwarded | Points awarded to the group this year. Points are awarded to the top 4 groups in descending order of cash on hand: 12, 9, 4, 1. |
| group | Group ID number |

**‘EndOfYearRankStatus’**: At the end of each year, info about each group is recorded.

|  |  |  |
| --- | --- | --- |
| *Variable* | *Description and Possible Values* | Sub variables |
| year | Year that is ending |  |
| allGroups | List of all the groups with info about each group. This is stored at the end of the year, before cash is zeroed but after points are allotted. There is one entry for each group in the game. | group, groupName  cash: cash at the end of the year  rank : rank compared to other  groups, based on total points  points: total points  marketValue: net worth of stocks at end of year |

**‘NewGameStart’**: Recorded at the start of each game.

|  |  |
| --- | --- |
| *Variable* | *Description and Possible Values* |
| startingYear | First year of the game |
| gameCode | Name of resource being cashed out |
| amount | Amount of resource being cashed out |

**Dictionaries**

The following dictionaries allow you to map item numbers onto item names, and map group IDs onto group names.

expRes = {"e1": "gold", "e2":"silver", "e3": "platinum", "e4": "diamond", "e5": "iridium", "e6": "ruby", "e7": "sapphire", "e8": "pearl"};

cheapRes = {"c1": "wood", "c2": "water", "c3": "food", "c4": "coal", "c5": "cats", "c6": "stone", "c7": "oil", "c8": "cotton"};

allRes = {"e1": "gold", "e2":"silver", "e3": "platinum", "e4": "diamond", "e5": "iridium", "e6": "ruby", "e7": "sapphire", "e8": "pearl", "c1": "wood", "c2": "water", "c3": "food", "c4": "coal", "c5": "cats", "c6": "stone", "c7": "oil", "c8": "cotton"};};

groups = {

1.0: 'red-group',

2.0: 'green-group',

3.0: 'pink-group',

4.0: 'blue-group',

5.0: 'yellow-group',

6.0: 'orange-group',

7.0: 'turquoise-group',

8.0: 'fuschia-group'

}

Zone Codes Map