Réflexions pour l’implémentation de Pijersi sur la plateforme Web Board Game Arena (BGA)

Table des matières

[1. Cahier des charges 1](#_Toc187568708)

[2. Etat stocké dans SQL 2](#_Toc187568709)

[3. GUI du joueur actif 3](#_Toc187568710)

[4. GUI du joueur passif 3](#_Toc187568711)

# Board Game Arena

# Software versions

Software Versions

Versions currently used by BGA framework:

• Dojo Toolkit 1.15

• PHP: 8.2

• SQL: MySQL 5.7

• JS/CSS/HTML: limited by what minimization tools support: JS minimization

• Font Awesome: 4.7 https://fontawesome.com/v4.7/icons/ (available as <i class="fa fa-clock" />)

• Font Awesome: 6.4.0 https://fontawesome.com/v6/search?o=r&m=free (available as <i class="fa6 fa6-clock" />)

PHP Extensions Used

The following PHP extensions are - as of May 8th, 2022 - in use in BGA Studio and available:

date, libxml, openssl, pcre, zlib, filter, hash, Reflection, SPL, session, standard, sodium, apache2handler, mysqlnd, PDO, xml, apcu, bz2, calendar, ctype, curl, dom, mbstring, FFI, fileinfo, ftp, gd, gettext, gmp, iconv, igbinary, json, exif, msgpack, mysqli, pdo\_mysql, apc, posix, readline, shmop, SimpleXML, sockets, sysvmsg, sysvsem, sysvshm, tokenizer, v8js, xmlreader, xmlwriter, xsl, zip, Phar, memcached, Zend OPcache

Javascript minimization (after July 2020)

For performance reasons, when deploying a game the javascript code is minimized using terser (https://github.com/terser/terser). This minifier works with modern javascript syntax. From your project "Manage game" page, you can now test a minified version of your javascript on the studio (and revert to the original).

NB: it has been reported that there is an issue with this minifier and percentage values for opacity.

<https://github.com/terser/terser>:

A JavaScript mangler/compressor toolkit for ES6+

# BGA architecture/protocole

* Début du jeu :
  + PHP : appel de « setupNewGame »
    - Create the right SQL statements to initialize the database according to the game rules
* Chargement du jeu :
  + PHP : appel de « getAllDatas »
  + JS :
    - Create some dynamic HTML page ; let the client do the game setup ;
    - The HTML generated embed automatically the data previously generated (in « json » format).
    - Load images, javascript and css stylesheet
    - Appel de « setup »
* Joueur actif :
  + JS : « OnClicMyMethod » envoie la requête par URL « /mygame/mygame/playAcard.html?card=99 »
  + PHP : « playAcard » est appelée
  + SQL : mettre à jour l’état du jeu
* Tous les joueurs :
  + « notifyAllPlayers » est renvoyé à tous les joueurs (actif ou pas).
  + La notification par un nom dédiée telle que « takeMoneyToken »
  + JS : la méthode « notif\_takeMoneyToken » est appelée avec des arguments (qui vient de jouer et quoi).
* Ensuite :
  + D’autres notifications sont envoyées :
    - Pour d’autres mises à jour des artéfacts du jeu
    - Et finalement, pour indiquer le nouveau « joueur courant ».

# Cahier des charges

* Pas d’IA
* Montrer les coups possibles
* Pas de demande de validation d’un coup.
  + Mais possibilité d’annuler un coup qui n’est pas complètement joué (comme dans l’implémentation Python)
* Montrer le même état du plateau à l’adversaire.
  + Même les états intermédiaires ; donc les hésitations de l’adversaire.
* Monter la trace du dernier coup joué par l’adversaire, comme dans l’implémentation Python.
* En partie différée, juste montrer la trace du dernier coup.
* Sur demande, montrer la liste des coups joués.
* Monter graphiquement les cubes capturés.

# Etat stocké dans SQL

|  |  |  |  |
| --- | --- | --- | --- |
| **Groupe/Table** | **Variable** | **Type/Domaine** | **Commentaire** |
| game\_state | game\_is\_terminated | Bool |  |
| (1 row) | game\_winner | Black/White/None |  |
|  | game\_credit | Int | game\_credit >= 0 |
|  | game\_turn | Int | game\_turn >= 0 ; game\_turn >= 1 dès que la partie a commencé |
|  | current\_player | Black/White/None |  |
|  | last\_player | Black/White/None | Le dernier joueur qui a joué |
|  | last\_turn | Int | Le numéro du tour déjà joué |
|  | last\_legend | String | La légende du dernier coup joué ; vide si game\_turn = 0 |
|  | last\_src\_hexagon | Int/None | La trace du dernier coup joué : position source |
|  | last\_int\_hexagon | Int/None | La trace du dernier coup joué : position intermédiaire |
|  | last\_dst\_hexagon | Int/None | La trace du dernier coup joué : position destination |
| **Groupe/Table** | **Variable** | **Type/Domaine** | **Commentaire** |
| last\_hexagon\_states | hexagon\_index | Int | **Key** (hexagon\_index >= 0) |
| (n rows) | hexagon\_position\_u | Int | Pourrait être codé en dur dans PHP |
|  | hexagon\_position\_v | Int | Pourrait être codé en dur dans PHP |
|  | hexagon\_label | String | Pourrait être codé en dur dans PHP |
|  | hexagon\_config | Void/Single/Stack |  |
|  | hexagon\_color | Black/White/None | Ce champ et les deux suivants pourraient se coder en une seule |
|  | hexagon\_bottom | Rock/Paper/Scissors/Wise/None | lettre : {‘R’, ‘P’, ‘S’, ‘W’, ‘r’, ‘p’, ‘s’, ‘w’, ‘\_’} avec ‘\_’ qui encode « vide » |
|  | hexagon\_bottom | Rock/Paper/Scissors/Wise/None |  |
| **Groupe/Table** | **Variable** | **Type/Domaine** | **Commentaire** |
| last\_captured\_cubes | cube\_color | Black/White | **Key** (part 1) |
| (n rows) | cube\_type | Rock/Paper/Scissors/Wise | **Key** (part 2) ; ou bien une clé compacte {‘R’, ‘P’, ‘S’, ‘W’, ‘r’, ‘p’, ‘s’, ‘w’} |
|  | cube\_count | Int | Value |
| **Groupe/Table** | **Variable** | **Type/Domaine** | **Commentaire** |
| turn\_legends | turn\_index | Int | **Key** ; turn\_index >= 0 ; ne pas afficher dans la liste des coups joués |
| (n rows) | turn\_legend | String |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Groupe/Table** | **Variable** | **Type/Domaine** | **Commentaire** |
| current\_hexagon\_states | hexagon\_index | Int | **Key** ; hexagon\_index >= 0 |
| (n rows) | hexagon\_position\_u | Int | Pourrait être codé en dur dans PHP |
|  | hexagon\_position\_v | Int | Pourrait être codé en dur dans PHP |
|  | hexagon\_label | String | Pourrait être codé en dur dans PHP |
|  | hexagon\_config | Void/Single/Stack |  |
|  | hexagon\_color | Black/White/None | Ce champ et les deux suivants pourraient se coder en une |
|  | hexagon\_bottom | Rock/Paper/Scissors/Wise/None | seule lettre : {‘R’, ‘P’, ‘S’, ‘W’, ‘r’, ‘p’, ‘s’, ‘w’, ‘\_’} avec ‘\_’ qui encode « vide » |
|  | hexagon\_bottom | Rock/Paper/Scissors/Wise/None |  |
| **Groupe/Table** | **Variable** | **Type/Domaine** | **Commentaire** |
| current\_captured\_cubes | cube\_color | Black/White | **Key** (part 1) |
| (n rows) | cube\_type | Rock/Paper/Scissors/Wise | **Key** (part 2) ; ou bien une clé compacte {‘R’, ‘P’, ‘S’, ‘W’, ‘r’, ‘p’, ‘s’, ‘w’} |
|  | cube\_count | Int | Value |

|  |  |  |  |
| --- | --- | --- | --- |
| **Groupe/Table** | **Variable** | **Type/Domaine** | **Commentaire** |
| turn\_state | action\_index | 1/2 | 1ère ou 2nde action |
| ( 1 row) | action\_src\_hexagon | Int/None | None ⬄pas de position sélectionnée |
|  | action\_src\_config | Single/Stack/Top | Top : le cube du haut d’une pile |
|  | action\_dst\_hexagon | Int/None | None ⬄pas de position sélectionnée |
|  |  |  |  |
|  | action\_choices | List[Int] | Liste calculée en PHP des cases sélectionnables |

* Les hexagones de « action\_choices » sont tous à cliquer et ils sont rendus :
  + en surbrillance claire au survol de souris si « action\_src\_hexagon == None » ou si « action\_dst\_hexagon == None »
  + en surbrillance foncée fixe si « action\_src\_hexagon != None  »
* La valeur « action\_index » pourrait être rendue dans le GUI, y compris celui de l’adversaire.

# GUI du joueur actif

* Minimum d’intelligence
* Trois zones :
  + Le plateau des hexagones
  + Les cubes capturés
  + La légende des coups déjà joués
* Deux boutons :
  + Pour montrer ou cacher la zone contenant la liste des coups déjà joués
  + Pour annuler le coup en préparation
* Seulement deux types d’événement sur les hexagones
  + **Survol** de certains hexagones : changement immédiat de l’affichage
  + **Click** de certains hexagones : renvoi de l’événement vers le serveur
* Piloté par l’état courant SQL :
  + Quoi afficher et comment
  + Quelles sont les hexagones dynamiques (associés aux deux types d’événements)

# GUI du joueur passif

* Aucune d’intelligence
* Réplication de l’état courant du plateau : après un coup joué ou même l’état intermédiaire
  + Uniquement les hexagones sélectionnés par l’adversaire
  + Pas les réactions au survol de la souris par l’adversaire.