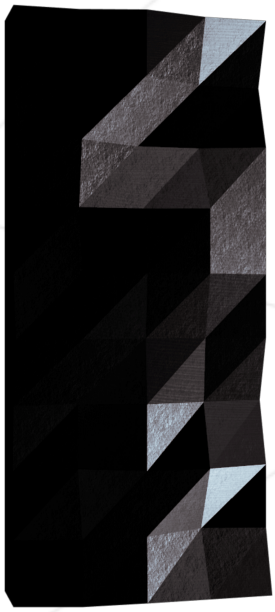

Formation Python-Django - Rush 00

English version coming soon !

Résumé: Ceci est le sujet du rush00 de la piscine Python-Django.

Table des matières

Chapitre I

Consignes

- Seule cette page servira de référence : ne vous fiez pas aux bruits de couloir.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Si aucune information contraire n'est explicitement présente, vous devez partir du principe que les versions des langages et framework utilisés sont les suivantes (ou ultérieures) :
  - `Python 3`
  - `HTML5`
  - `CSS 3`
  - `Javascript EM6`
  - `Django 1.9`
  - `psycopg2 2.6`
- Sauf indication contraire dans le sujet, les fichiers en python de chaque exercice sur `Python` seul (d01, d02 et d03) doivent comporter à leur fin un bloc `if __name__ == '__main__'`: afin d'y insérer le point d'entrée dans le cas d'un programme, ou des tests dans le cas d'un module.
- Sauf indication contraire dans le sujet, chaque exercice des journées portant sur `Django` aura sa propre application dans le projet à rendre pour des raisons pédagogiques.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne porterons attention ni ne prendrons en compte un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Attention aux droits de vos fichiers et de vos répertoires.
- Vous devez suivre la procédure de rendu pour tous vos exercices : seul le travail présent sur votre dépot GIT sera évalué en soutenance.
- Vos exercices seront évalués par vos camarades de piscine.
- Vous ne devez laisser dans votre répertoire aucun autre fichier que ceux explicitement specifiés par les énoncés des exercices.
- Sauf indication contraire dans le sujet vous ne devez pas inclure dans votre rendu :

- Les dossiers `__pycache__`.
- Les éventuelles migrations.
  Attention, il vous est tout de même conseillé de rendre le fichier `migrations/__init__.py`, il n'est pas nécessaire mais simplifie la construction des migrations.
  Ne pas ajouter ce fichier n'invalidera pas votre rendu mais vous *devez* être capables de gérer vos migrations en correction dans ce cas.
- Le dossier créé par la commande `collectstatic` de `manage.py` (avec pour chemin la valeur de la variable `STATIC_ROOT`).
- Les fichier en bytecode Python (Les fichiers avec une extension en `.pyc`).
- Les fichiers de base de donnée (notamment avec `sqlite`).
- Tout fichier ou dossier devant ou pouvant être créé par le comportement normal du travail rendu.

Il vous est recommandé de modifier votre `.gitignore` afin d'éviter les accidents.

- Lorsque vous devez obtenir une sortie précise dans vos programmes, il est bien entendu interdit d'afficher une sortie précalculée au lieu de réaliser l'exercice correctement.
- Vous avez une question ? Demandez à votre voisin de droite. Sinon, essayez avec votre voisin de gauche.
- Votre manuel de référence s'appelle `Google / man / Internet / ...`.
- Pensez à discuter sur le forum Piscine de votre Intra !
- Lisez attentivement les exemples. Ils pourraient bien requérir des choses qui ne sont pas autrement précisées dans le sujet...
- Par pitié, par Thor et par Odin ! Réfléchissez nom d'une pipe !

Chapitre II

Règles spécifiques de la journée

- L'interprêteur à utiliser est `Python3`
- Les routes relatives à une application doivent être définies dans un fichier `urls.py` se trouvant dans le dossier de cette application.
- Chaque page affichée doit être correctement formatée (présence d'un doctype, de couples de balises `html`, `body`, `head`), gestion correcte des caractères spéciaux, pas d'affichage bizarre.
- Le serveur utilisé pour ce rush est le serveur de développement par défaut de Django fourni avec l'utilitaire `manage.py`.
- Seules les URLs explicitement demandées doivent retourner une page sans erreur. Ainsi, si seule `/ex00` est demandée, `/ex00foo` doit retourner une erreur 404.
- Vous devez rendre un fichier requirement.txt (via pip freeze) contenant les librairies nécessaire au fonctionnement de votre projet.

Chapitre III

Préambule

Some quotes from the movie Be Kind Rewind with Jack Black :

FIGURE III.1 – The video store.

Jerry : *I am Robocop. Anything you say can and will be held against you in a court of Robocop.*

Jerry : *I will shoot you, and I know robot karate.*

Jerry : *I'm not saying your uncle is illerate, maybe he just needs to go to nightschool.*

Jerry : *I was going to make you into the next Marylin Monroe ! But no ! Now you're just going to be.... Laundry girl !*

Jerry : *Hey, hey, don't put your shoes in the refrigerator... 'cause they'll get cold !*

Mike : (holding the Ghost Busters tape) *I'll be Bill Murray and you'll be everyone else.*

Jerry : *[sung, poorly, to the tune of the Ghostbusters theme song] When you're walkin' down the street...*

Jerry : *[singing] ... and you see a little ghost...*

Jerry : *[singing] ... whatcha gonna do about -*

Jerry : *Ghostbusters ?*

Mike : *What ? What is that ?*

Jerry : *That's the Ghostbusters theme song.*

Mike : *No.*

Jerry : *I'm pretty sure it is.*

Alma : *Are you in love with me ?*

Mike : *Huh ?*

Alma : *Mm-hm.*

Mike : *Well, how do I know that ?*

Alma : *You know you're in love with a person when you talk to them for a minimum of 20 minutes a day in your head.*

Mike : *What if I talk to a guy in my head for 20 minutes ? What would that mean ?*

Alma : *You're in love with Jerry.*

Your name it, we shoot it

Sometimes the best movies are the ones we make up

source

Chapitre IV

Partie obligatoire

IV.1 Introduction - FAQ

Ce rush à pour objectif de vous faire coder un petit jeu solo doté d'une interface Web.

`Quel est l'objectif de ce jeu ?`
Ce jeu se nomme `MovieMon` et son but est de capturer tous les Moviemons qui se cachent sur une grille de jeu en se servant de Movieballs.

`Ça me dit quelque chose, ça ne ressemblerait pas à ... ?`
Je ne vois pas de quoi vous voulez parler.

`Qu'est ce qu'un Moviemon ?`
Un `Moviemon` est un film disponible sur IMDB. Idéalement un film de monstre.

`Comment attrappe-t-on un Moviemon ?`
En lui jetant une `movieball` voyons !. Le rating imdb d'un `Moviemon` correpond à sa force. Un rating élevé rend le `Moviemon` plus difficile à attrapper qu'un rating bas. La force du joueur, qui correspond au nombre de `Moviemons` en sa possession, augmente ses chances d'en attrapper.

`Comment se déroule une nouvelle partie typique ?`
Lorsqu'un joueur démarre une nouvelle partie, le jeu requête sur IMDB tous les films nécéssaires avant de l'envoyer sur la 'Worldmap', la page principale du jeu.
Sur cette page, le joueur se déplace librement et allègrement de case en case, sur une grille de taille fixe. Au hasard des cases sur lesquelles il marche, il récolte des `movieballs`, ou débusque un `Moviemon`.
Lorsqu'il débusque un `Moviemon`, s'il estime avoir ses chances, le joueur tente de le capturer.
Il lui lance une movieball, rate, une deuxième, rate, une troisième, attrapé ! Le joueur consulte alors fièrement son `Moviedex` qui répertorie tous les `Moviemons` capturés, avant de repartir en chasse pour tous les attrapper ! !

IV.2 Consignes

IV.2.1 Settings

Vous devez ajouter aux settings de votre projet ceux du jeu en lui-même, à savoir :

- La taille de la grille. Celle-ci doit faire un minimum de dix cases de haut et dix cases de large.
- La position de départ du joueur dans cette grille
- La liste des noms ou ids d'au moins dix films requêtable sur IMDB.

Vous devez utiliser ces settings dans votre code.

Vous pouvez rajouter d'autres settings si vous le souhaitez.

Le rating d'un film corresponds à sa force. Ainsi pensez à équilibrer votre jeu en prenant suffisamment de bons et de mauvais films. Par exemple : au moins trois avec score inférieur à quatre et trois avec score supérieur à sept

L'API d'IMDB étant plutôt obscure, vous pouvez utiliser OMDB, qui est une API officieuse.

IV.2.2 Données de jeu

Vous allez avoir besoin de conserver des données entre les différentes pages lors d'une scéance de jeu. Un site web typique utiliserait des cookies, ou encore un système de session coté serveur. Mais ici il n'est pas question de site web typique.

Vous devez stocker les données résumant l'état de la partie en cours dans un fichier qui doit être créé dans votre projet.

Ce fichier n'est pas une sauvegarde pour le joueur, il sert à stocker l'état du jeu en cours. Il ne doit contenir aucune logique et son contenu doit être stocké en binaire grace à la librairie `pickle` (incluse avec `Python`).

Vous devez donc également créer dans votre projet la logique nécessaires à la mise à jour et à l'utilisation de ce fichier qui doit contenir les informations suivantes :

- La position du joueur sur la map.
- Le nombre de `Movieballs` possédées.
- Les noms (ou identifiants) de tous les `Moviemons` du `Moviedex`.

- Les informations complêtes de tous les `Moviemons` de la partie, tels qu'obtenus sur `IMDB`.

IV.2.3 Gestion des données

Vous devez également créer une classe `Python` qui a pour mission de gérer ces données de jeu. Cette classe doit à minima contenir les méthodes suivantes :

- 'load' : Charge les données de jeu données passés en paramètres dans l'instance de classe. Retourne l'instance courante.
- 'dump' : Retourne les données de jeu.
- 'get_random_movie' : Retourne un Moviemon au hasard parmis les `Moviemons` non capturés.
- 'load_default_settings' : Charge les données de jeu dans l'instance de classe depuis les settings. Requête et stock les détails de tous les `Moviemons` sur `IMDB`. Retourne l'instance courante.
- 'get_strength' : Retourne la force du joueur.
- 'get_movie' : Retourne un dictionnaire `Python` contenant tous les détails du nom de `Moviemon` passé en paramètre et nécessaires à la page `Detail`.

Vous pouvez ajouter à cette classe autant de méthodes et d'attributs que nécessaire.

`Il ne sera jamais demandé de modifier "à la main" ou de supprimer ce fichier ou n'importe quel fichier de sauvegarde en cour de jeu en soutenance.`

IV.2.4 Esthétique du jeu

L'affichage du jeu se fera naturellement sur votre navigateur via du `HTML` et du `CSS`. Vous n'avez pas la permission d'utiliser de `Javascript`.

L'affichage du jeu est scindé en deux parties qui doivent être visuellement parfaitement distinguable :

- `L'écran` : Affiche ce qui se passe dans le jeu. Aucune interaction n'est possible à cet endroit qui ne doit jamais contenir ni lien ni aucun formulaire.

- `Les contrôles` : Situés en dessous ou de part et d'autre de l'écran, ils permettent d'interragir avec le jeu et sont contextuels, c'est à dire qu'ils changent de comportement en fonction de l'état du jeu. Cela signifie également qu'ils ne sont pas nécessairement tous actifs systématiquement. En revanche, même inactifs, ils doivent être `visibles et garder la même place en permanence`.

Il doit y avoir huit 'boutons' :

- Quatre `directions`, comme celles qu'on pourrait trouver sur une une croix directionnelle de manette de jeu :
  Haut, droite, bas, gauche
- Un bouton `select`.
- Un bouton `start`.
- Un bouton `A`
- Un bouton `B`

Vous n'avez pas la permission de rajouter/supprimer de 'boutons' ni d'afficher d'informations, en dehors du nom des 'boutons', dans cette zone.

Au delà de cette distinction minimum, l'esthétique n'a pas d'importance dans la partie obligatoire du sujet.

Le comportement des boutons pour chaque vue est décrit dans la section suivante.

`Un 'bouton' n'est pas nécessairement un tag HTML <bouton>. Un 'bouton' inactif peut être un lien mort, une image, un texte, ou un caractère spéciale.`

IV.2.5 Les pages

Vous devez créer les pages/vues/comportements listées si après. Un 'bouton' non mentionné dans une page est un 'bouton' inactif. De plus, si la destination n'est pas précisé pour un contrôle, c'est que celui-ci renvoye la même page, potentiellement modifiée.

`TitleScreen`

- Description : Ecran d'accueil
- Ecran : Doit afficher le nom du jeu ainsi que 'A - New Game' et 'B - Load'.
- Url : celle de base, nom de domaine plus port.
- Controles :
  - `A` : Lien vers la page `Worldmap`.
    Avant d'afficher celle-ci, le fichier contenant les informations de la partie en cours doit être réinitialisé avec les paramêtres des `Settings` et les `Moviemons` doivent être requêtés à nouveau.
  - `B` : lien vers la page `Load`

`Worldmap`

- Description : carte du jeu, où le personnage se déplace, récupère des `movieballs` et débusque des `Moviemons`.
- Ecran : Une grille dont la taille est celle définie dans les settings. Sur la case correspondant à la position actuelle du joueur doit se trouver une représentation (image, caractère, etc ...) clairement identifiable du personnage.

  L'écran doit également afficher :

  - Le nombre de `movieballs`.
  - Un message lorsqu'une `movieball` est trouvée.
  - Un message lorsqu'un `Moviemon` est débusquée ainsi qu'une indication du bouton sur lequel appuyer pour entamer la capture.]
- Url : `'/worldmap'`
- Contrôles :
  - `Directions` : Chaque direction doit déplacer le personnage d'une case dans la même direction. Le joueur ne doit pas pouvoir sortir de la map.

    Chaque déplacement a une chance de débusquer un `Moviemon` ou de récolter une `movieball`.

    Si un `Moviemon` est débusqué, celui-ci est choisit au hasard parmi les `Moviemons` encore non capturés.

  - `A` : Uniquement si un `Moviemon` est débusqué : Lien vers la page `Battle` du combat contre ce `Moviemon`.
  - `start` : lien vers la page `Option`.
  - `select` : lien vers la page `Moviedex`.

`Rafraîchir cette page ne doit pas modifier la position du personnage sur la carte.`

`Battle`

- Description : Essayez de capturer le Moviemon que vous avez débusqué !
- URL : `'/battle/<moviemon_id>'`. <moviemon_id> est à remplacer par l'identifiant du `Moviemon` à combattre, quel qu'il soit.

- Ecran : Affiche le poster et la force du `Moviemon`, le nombre de `movieballs` en stock, la force du joueur ainsi que le taux chance de réussite (cf plus bas).

  En cas de capture, vous devez également afficher une phrase du type "You caught it" pour marquer l'evenement.

  En cas d'echec d'un lancé, vous devez afficher de la même façon une phrase du type "You missed !"".

  Tant que le `Moviemon` n'est pas capturé, l'écran doit également afficher 'A - Launch movieball'.

  Dans tous les cas, vous devez afficher que le 'bouton' `B` ramène à la `Worldmap`.

- Contrôles :
  - `A` : Lance une `movieball`

    Si le joueur n'en a pas, l'action n'a aucun effet (vous pouvez afficher sur l'écran une phrase de moquerie de la part de l'ennemi).

    Dans le cas contraire, le nombre de movieball est diminuée d'un et un jet de chance est fait pour savoir si le `Moviemon` est capturé ou non.

    Le taux de chance `C` est calculé de la façon suivante :
    `C` = 50 - (force du monstre * 10) + (force du joueur * 5)
    Et 1 <= `C` <= 90

    Par exemple :
    Pour un monstre de puissance 8.2 et un joueur de force 2 :
    C = 50 - 82 + 10 = -22
    Ce monstre a donc 1% de chance de s'être attrapé.

    Pour un monstre de puissance 5 et un joueur de force 8 :
    C = 50 - 50 + 40 = 40
    Ce monstre a donc 40% de chance d'être attrapé.

    Pour un monstre de puissance 2 et un joueur de force 14 :
    C = 50 - 20 + 70 = 100
    Ce monstre a donc 90% de chance d'être attrapé.

    En cas de succès, le `Moviemon` est capturé puis stocké dans le `MovieDex`. Le bouton A devient alors inactif.

    En cas d'echec, le joueur peut relancer autant de movieballs dont il dispose.

  - `B` : Retour à la page `Worldmap`

`Moviedex`

- Description : Liste des `Moviemons` capturés. Il doit être possible de sélectionner un film pour accéder aux détails de celui-ci.
- URL : `'/moviedex'`
- Ecran : Doivent être présent à l'écran tous les posters des `Moviemons` capturés. L'affiche sélectionnée doit être marqué d'un élément graphique distinctif, par exemple en l'encadrant d'une bordure bleue.

  Vous devez également rajouter 'A - More information' et 'select - Back'.
- Contrôles :
  - `Directions` : Les directions permettent de sélectionner un film différent. Vous devez utiliser au moins deux directions : gauche et droite ou haut et bas.
  - `select` : Lien vers la page `Worldmap`
  - `A` : Lien vers la page `Detail` du `Moviemon` sélectionné.

`Par défaut en arrivant sur la page, le premier film de la liste est sélectionné.`

`Detail`

- Description : Détail d'un moviemon.
- URL : `'/moviedex/<moviemon>'` <moviemon> doit être remplacé par l'identifiant du `Moviemon`.
- Ecran : Doit afficher le nom, le poster, le realisateur, l'année, le rating, le synopsis et les acteurs du Moviemon ainsi que 'B - Back'
- Controles :
  - Bouton B : Lien vers la page Moviedex

`Option`

- Description : Les options du jeu.
- URL : `'/options'`
- Ecran : Affiche les options de jeu, qui sont 'A - Save', 'B - Quit' ainsi que 'start - cancel'
- Contrôles :
  - Bouton start : lien vers la page `Worldmap`.
  - Bouton A : Lien vers la page `Save`.

- Bouton B : Lien vers la page `TitleScreen`.

`Save`

- Description : Permet de sauvegarder une partie en cours dans un parmis trois slots disponibles.
- URL : `/options/save_game`
- Ecran : doit afficher trois slots : 'Slot A', 'Slot B' et 'Slot C'.

  Un slot sélectionné doit être marqué d'un élément graphique distinctif, par exemple en le faisant précéder d'une flêche.

  Si un slot est vide, il doit être suivit de 'Free'. Dans le cas contraire, il doit être suivit du nombre de `Moviemons` capturés.

  Par exemple : 'Slot A : 2/15' signifie que cette sauvegarde contient une partie dans laquelle deux `Moviemons` sur quinze ont été capturés.

  L'écran doit également afficher 'A - Save' et 'B - Cancel'
- Contrôles :
  - `Directions` : Les directions 'haut' et 'bas' doivent servir à selectionner un slot.
  - `A` : Copie le fichier contenant l´état actuelle du jeu dans un autre fichier qui doit être stocké dans un dossier 'saved_game' dans votre projet.

    Le nom de ce fichier doit être 'slot<n>_<score>.mmg'. <n> est a remplacer par le nom du slot, 'a', 'b' ou 'c' et <score> est a remplacer par le score de la partie.
    Par exemple : slotb_1_15.mmg. (S'il vous plait, n'essayez pas de mettre un slash dans un nom de fichier).
  - `B` : Lien vers la page `Option`

`Load`

- Description : Permet de charger une partie sauvegardé parmis trois slots.
- URL : `/options/load_game`
- Ecran : doit afficher trois slots : 'Slot A', 'Slot B' et 'Slot C'.

  Un slot sélectionné doit être marqué d'un élément graphique distinctif, par exemple en le faisant précéder d'une flêche.

  Si un slot est vide, il doit être suivit de 'Free'. Dans le cas contraire, il doit être suivit du nombre de `Moviemons` capturés.

Par exemple : 'Slot A : 2/15' signifie que cette sauvegarde contient une partie dans laquelle deux `Moviemons` sur quinze ont été capturés.

L'écran doit également afficher 'A - Load' et 'B - Cancel'.
Une fois qu'une partie est chargée 'A - Load' doit être remplacé par 'A - start game'

- `Directions` : Les directions 'haut' et 'bas' doivent servir à selectionner un slot.
- `A` :
  Si aucune partie n'est chargé : Copie le contenu du fichier correspondant au slot sélectionné dans le fichier servant à stocker l'état actuel du jeu.

  Si une partie vient d'être chargée, le 'bouton' sert de lien vers la page `Worldmap`.

  Tenter de charger un slot 'Free' ne doit évidemment avoir aucun effet.
- `B` : Lien vers la page `TitleScreen`

Chapitre V

Partie Bonus

Une fois votre partie obligatoire parfaitement réalisée vous pouvez implémenter des fonctionnalités supplémentaires afin de gagner des points bonus.

Pour que votre correcteur considère ces fonctionnalités comme réussies, vous devrez le convaincre de leur bonne réalisation.
Une erreur non gérée invalidera la fonctionnalité en question.

Uniquement dans le cadre des bonus, l'utilisation de Javascript est tolérée tant que celle-ci n'altère pas le fonctionnement de la partie obligatoire (qui doit fonctionner sans javascript). L'AJAX ou les Websockets ne sont pas autorisés.

Voici quelques idées de bonus que vous pourriez implémenter :

- Faites ressembler votre site à une console de jeu (cf exemples) :
  - L'écran doit avoir l'air d'etre un véritable écran.
  - Les controles de votre jeu doivent être superposés à des images de véritable contrôles imitant une console de jeux.
- Dans l'état actuel du jeu, il est parfaitement possible de deviner l'adresse d'un combat et d'attrapper les monstres sans avoir à les chercher.
  Créez un système de token difficile voir impossible à deviner, associés à des `Moviemons`, à utiliser dans l'url afin de prévenir la triche.
- Associez les contrôles à des touches du clavier afin de ne pas avoir à utiliser la souris pour jouer.
- Afin de varier le jeu, faites que chaque nouvelle partie charge une sélèction différente de monstres.
- Ajoutez de la variété à la `Worldmap` avec un radar qui affiche les `Moviemons`/`moviballs` disponibles sur les cases adjacentes à la position du joueur.

Chapitre VI

Rendu et peer-évaluation

Vous devez rendre un projet `Django` parfaitement configuré.

Hormis de ce qui vous est imposé dans le sujet, vous êtes libre d'organiser ce projet comme vous l'entendez.

Aucune erreur serveur ne sera toléré. Testez bien les comportements de vos vues.

Vous devez fournir un fichier requirement.txt contenant toutes les librairies nécessaires au fonctionnement de votre projet.

`Django renvoie automatiquement une page 404 si l'url entrée est introuvable. Vous pourriez tomber sur des cas ou Django trouve une url qui corresponds, mais qui dans votre logique est une erreur. Dans ce cas, pensez à utilisez "raise Http404('my message')" dans vos vues.`

Chapitre VII

Exemple de rendu

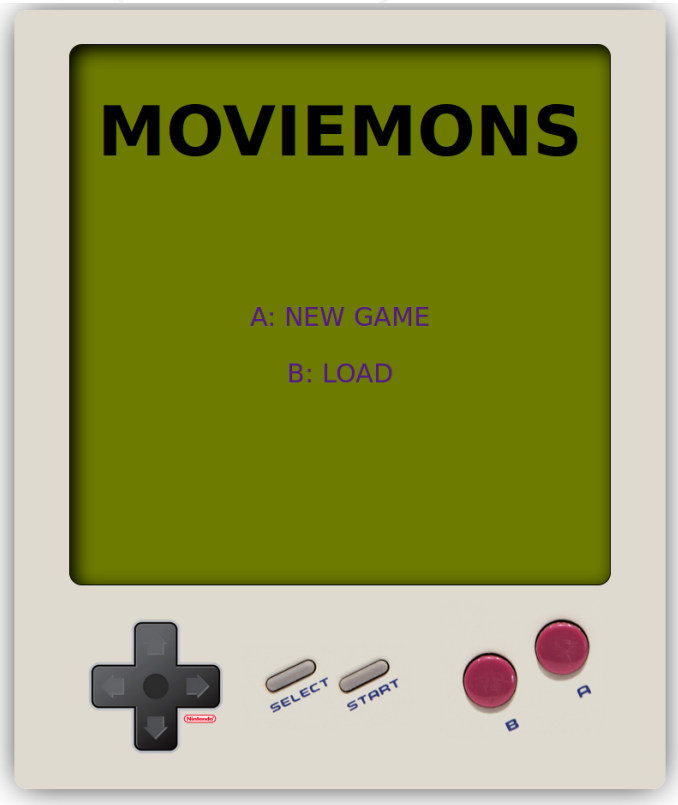

FIGURE VII.1 – Your titlescreen could look like that

FIGURE VII.2 – This Moviemon game appears to happen in Belgium