**Comment fonctionne le nouveau moteur physique ?**

Nom de la méthode : impulse based dynamics.

Idée principale : on ne calcule jamais les forces explicitement, on intègre immédiatement les vitesses puis on corrige ces vitesses pour respecter des contraintes (⬄calculer les forces de réaction).

Exemple du pendule : Calcul des forces exterieures (pesanteur), Correction des vitesses pour qu’elle soit // à la tige, correction de la position via la vitesse pour garantir que la tige ne change pas de taille.

Effet des différentes variables globales :

* dt (stabilité vs temps de calcul)
* Fps (fluidité vs temps de calcul)
* g (à régler pour que ce soit réaliste)
* quelques variables locales (=> passer en global ???) qui change des choses (coeff, seuil…)

Pistes d’amélioration :

* Correction des positions sans passer par la vitesse (intégrer immédiatement) permet davantage de stabilité
* Enlever les effets « collage » qui arrivent parfois (= checker en + si les objets s’éloignent)
* Ajouter l’effet des corrections de position sur la vitesse angulaire
* Conditions de stabilité pour arrêter de calculer des forces/collisions
* Faire un vrai solveur avec inégalités ??? (compliqué…)

**Idée pour l’écran de démarrage :**

Apparition du nom du jeu de manière un peu random…

* Choix d’une initialisation de génération aléatoire
* Calcul des trajectoires
* Choix des couleurs
* Le nom du jeu apparaît par magie !!!

**Fonctions / Classes à modifier :**

* Spring => faire en sorte que l’on puisse attacher le ressort à n’importe quel endroit d’un rectangle/disque
* Vehicule + Weapons à inclure dans la boucle principale