Baba Is You

Analyse théorique :

1. De quel type de jeu s'agit-il ?
« Baba is You » est un puzzle game en vue de dessus.

2. Après une heure de jeu, nommez les différentes mécaniques de gameplay rencontrées.
Le joueur peux déplacer son (ses) personnages dans les quatre directions du clavier, et peut pousser des objets (qui ont étés préalablement marqués comme poussable) et des mots en allant vers eux à leur contact.
Les mots, lorsqu'ils forment une phrase, permettent de changer les règles du jeu, comme par exemple changer l'objet de fin de niveau, détruire un objet qui devrait te tuer, ou encore changer ce qu'on contrôle.

3. Ce jeu est considéré comme un jeu fait pour des développeurs, expliquez en quoi.
Le jeu ne gagnera pas de titres basés sur ses graphismes, qui sont simplistes au possible, mais son gamefeel a été poussé au maximum, et ses mécaniques perfectionnées le plus possible.

4. L'association des mots se font selon certaines règles :
"Baba Is Wall" -> valide /
"Wall Is You" -> valide /
"You Is Baba" -> non valide /
"Baba and Wall Is You" -> valide
Expliquez comment ces règles fonctionnent.
Il y a trois types de mots :
- les mots objets, qui servent de sujet et permettent de changer le sprite et ce qu'il fait
- les mots de liaison, qui permettent de former les règles et sont forcément placés entre les deux autres types d'objets
- les mots d'action, qui servent à associer une action à un sprite.

Un objet peut être un autre objet, et un objet peut effectuer une action. Cependant, une action ne peut pas être un objet ou une autre action.

5. Vu le nombre de combinaisons possibles, d'après vous, comment le développeur a-t-il fait pour obtenir un tel résultat ? Expliquez en termes techniques (pseudo-code, algorithmes, type de données, modifications du gameplay…) comment vous auriez mis en place une telle mécanique de gameplay.
J'aurais codé un objet qui prend plusieurs paramètre de départ, pour définir quel mot il affiche ; et dépendamment du mot, dans le begin play, je lui aurais changé un booléen qui définit si il est un objet, une liaison, ou une action.
Une fois fait, je lui aurais mis des triggers sur les cases adjacentes en bas et à droite, qui détectent tous les autres objets qui sont de son type. Le mot objet cherche le type d'objet à côté de lui, et si il est acceptable (liaison après un objet, action ou objet après une liaison …) prend son type et effectue une action spécifique.
Par exemple, « Baba is Wall » : « Baba » regarderait le « is », qui lui regarderait le « Wall ». Etant un mot objet, « is » lancerait un changement de sprite/static mesh sur l'objet « Baba », prenant le static mesh/sprite de « wall ».

6. Personnel : le jeu vous a-t-il plu ? Pourquoi ?
Le concept est original, et les puzzles sont courts, funs, et amusants, tout en restant de vrai casses-têtes.

Pratique :

1. Définissez, selon vous, la mécanique de gameplay principale du jeu, et reproduisez là sous le moteur de jeu de votre choix (vous pouvez faire cette mécanique sur un projet en 2D ou 3D).

La mécanique principale du jeu est le déplacement case par case du joueur et des objets qui sont devant lui. L'entièreté du jeu est présenté sous cette optique de « déplacer un personnage pour déplacer des objets », et même si l'aspect des mots qui changent les règles sont la seconde mécanique principale du jeu, je considère qu'elle est adjacente et sujette aux déplacements susnommés.

2. Décrivez, étape par étape, comment vous avez mis en place cette mécanique.

J'ai créé un objet principal « BP_Pushable », qui sert à la fois de joueur et d'objet à pousser, et si j'en avais été capable, d'objets mots (qui existent actuellement mais cachés sous forme de cubes avec du texte et sans mécaniques).

J'ai ensuité caché le main character, et supprimé ses déplacements ; au begin play, il récupère tous les acteurs de la classe « BP_Pushable », et lorsque le joueur appuie sur une touche de déplacement, tous les BP_Pushables qui sont actuellement notés comme « joueur » reçoivent le type de déplacement.

Le « BP_Pushable » possède quatre box collisions, qui lui permettent de détecter si, si il venait à être poussé dans une direction, il rentrerait en collision avec un autre objet. L'objet sera alors déplacé dans la même direction, répétant la vérification jusqu'à ce que soit la case suivante soit vide, soit un mur immovable. Dans le cas du mur immovable, aucun objet ne bougera.

Enfin, un BP_Caméra a été créé et placé en l'air, regardant le sol ; le BP_Gamemode, quand à lui, récupère la caméra au begin play et la set-up en temps que caméra de vue principale.

3. Expliquez en quoi cette mécanique est primordiale tout au long du jeu.

Si le joueur ne peut rien déplacer, et/ou ne peux pas bouger, il ne peux pas régler les problèmes et les énigmes qui lui sont présentés.

4. Pourquoi cette mécanique vous a-t-elle plu ?

Car elle est simple, efficace, et permet un grand nombre de types de puzzles différents en ajoutant quelques règles.