Cahier des charges *Terminal 20*

ÉCOLE NATIONALE D'INGÉNIEURS DE BREST

Auteur	Date	Version
Romain MELLAZA	21/02/2025	v0.2

Sommaire :

Objectifs

Description générale

Terminal 20 est un jeu vidéo réalisé dans le cadre du module "*Initiation Projet Informatique*" en semestre 2 à l'ENIB (*École nationale d'ingénieurs de Brest*). Ce document constitue le cahier des charges pour l'application en question.

Contexte et contraintes

Le jeu proposé ici sera minimaliste, que ce soit au niveau du design ou de l'architecture matérielle nécessaire. Il doit par ailleurs être totalement portable pour que l'expérience utilisateur soit la plus fluide possible.

Expression du besoin

Synopsis

Le jeu sera basé sur l'univers de *science-fiction*, en profitant notamment de l'ASCII-Art pour réaliser des effets visuels propre au film *Matrix* par exemple.

Le personnage incarné par le joueur sera un agent de la *DGSV* (Direction Générale de la Sécurité Virtuelle). Un matin, il ouvre sa messagerie professionnelle et tombe sur un mail menaçant qui l'incite à télécharger un exécutable, le jeu se lance alors, il a 20 minutes pour le finir sous peine qu'une bombe explose en plein centre-ville de Brest.

SPOILER : Dans la dernière salle du jeu se trouve un petit garçon, c'est Timmy, c'est lui qui a codé le jeu mais il est triste que personne n'y joue, il a donc monté de toute pièce cette histoire de bombe et envoyé ce mail à la DGSV pour se faire connaître.

Règles du jeu

Le jeu consiste en un labyrinthe où **il ne faut surtout pas toucher les murs**. Des personnages et des monstres seront présents pour aider, ou au contraire ralentir le joueur dans sa progression. Des énigmes devront être résolues par le joueur. Il s'agit donc d'un escape game virtuel.

Le joueur dispose d'exactement 20 minutes pour terminer le jeu. Il pourra consulter le temps restant à tout moment dans le coin supérieur gauche de son écran.

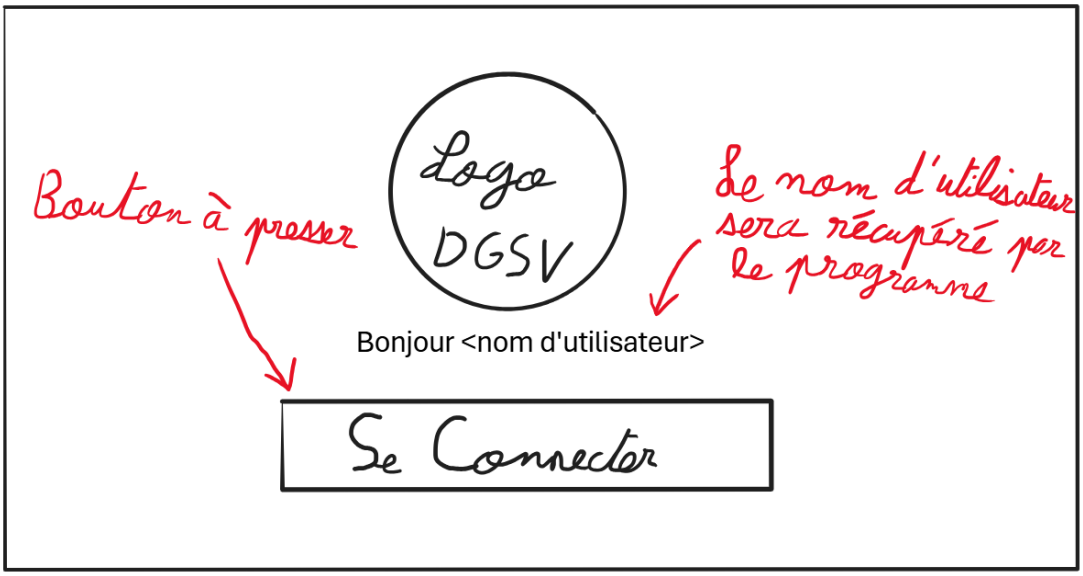
Les personnages rencontrés durant la partie questionneront le joueur, plusieurs réponses lui seront alors proposées, s'il choisit la bonne réponse, l'indication fournie en récompense sera juste et permettra de faire le bon choix pour la suite du parcours, sinon, l'indication sera erronée et pourrait faire perdre un temps précieux au joueur s'il fait malheureusement confiance à cette pénalité. En effet, toute la complexité du dilemme réside dans le fait que **le joueur ne saura pas s'il a bien ou mal répondu.**

Le 3ème personnage rencontré proposera au joueur 4 objets différents. Le joueur devra alors choisir l'arme qu'il considère comme la plus adaptée au combat qu'il va mener. La complexité du choix réside dans le fait que **le joueur ne connaît pas le monstre qu'il va affronter**.

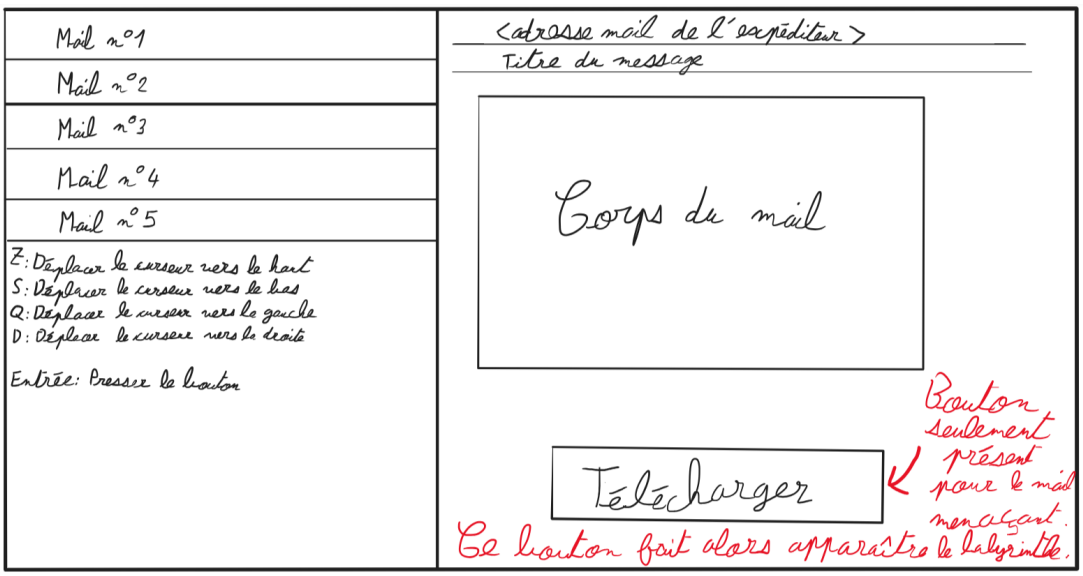
Interfaces utilisateur

Introduction

- Étape 1 : Le joueur vient d'exécuter la commande `python terminal20.py`. Après quelques lignes de commandes qui se déroulent rapidement à l'écran, cette page de connexion s'affiche.

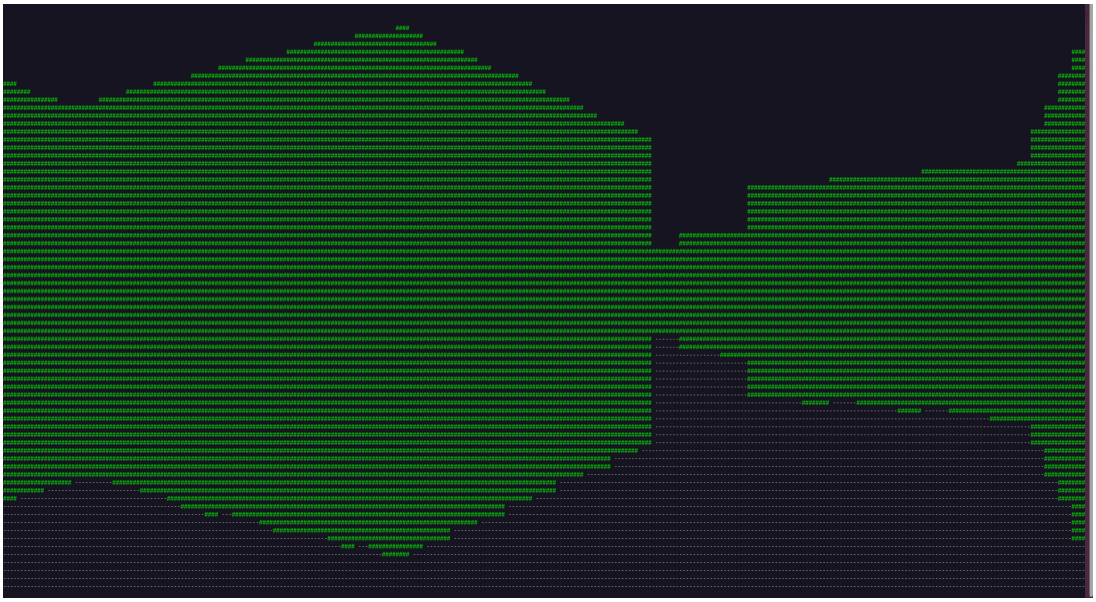


- Étape 2 : Une *command line interface* de boîte mail s'affiche à l'écran, 5 mails sont disponibles, 4 d'entre eux sont sans importance, cependant le 3ème est la mail contenant le bouton permettant de lancer réellement la partie. On peut représenter cette interface de cette façon :



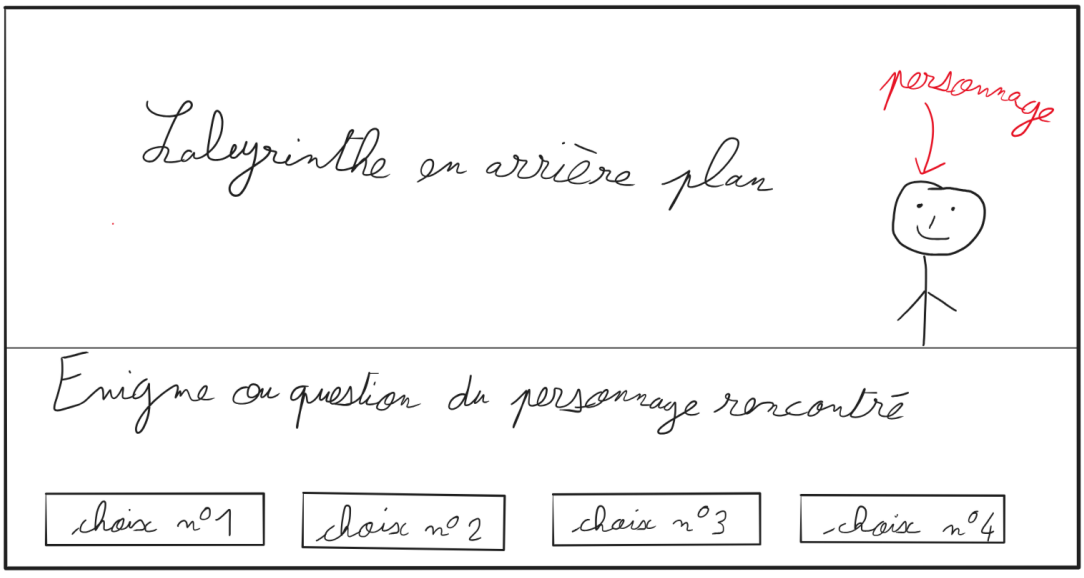
Labyrinthe

Il s'agit du rendu obtenu grâce à mon moteur graphique ASCII (au moment où je rédige ce document). Des améliorations de texture/coloration pourront être réalisées dans les prochaines versions du service.



Dialogues

Chaque interaction avec un personnage donne lieu à une courte conversation, puis le joueur doit faire un choix entre quatres propositions.



Manuel utilisateur

Commandes

- **Z** : avancer, sélectionner le mail supérieur
- **S** : reculer, sélectionner le mail inférieur
- **Q** : se tourner vers la gauche, déplacer le curseur vers le corps du message
- **D** : se tourner vers la droite, déplacer le curseur vers la zone de listing des messages

- `Enter` : Interagir (avec boîte de dialogues, objet/arme)
- `Echap` : Quitter le jeu

Personnages

- Personnage n°1 :
    - Nom : Chuck
    - Position dans le labyrinthe : Début
    - Particularité : Ce personnage a pour rôle principal d'expliquer de manière très succincte le but du jeu ainsi que les règles importantes. Il propose tout de même une énigme à la fin de son monologue.
    - Apparence : Il a une casquette, une hache à la main et un T-Shirt "make ENIB great again".
    - Énigme : "Il me faut 1min32s pour couper une bûche en deux. Combien de temps me faut-il pour couper une bûche en 16 morceaux de même taille ?"
        - Réponse A : 13 minutes
        - Réponse B : 23 minutes
        - Réponse C : 11,5 minutes
        - Réponse D : 18 minutes
- Personnage n°2 :
    - Nom : Madeleine
    - Position dans le labyrinthe :
    - Apparence : Madeleine est une bibliothécaire avec des petites lunettes rondes sur le bout du nez. Elle porte une pile de livres dans la main droite.
    - Énigme : Sur une étagère, j'ai rangé des livres faisant 48mm et d'autres faisant 32mm. L'étagère qui mesure 244,8cm est complètement pleine. Combien ai-je rangé de livres de 48mm ? (toutes les réponses proposés sont en base 13)
        - Réponse A : 42
        - Réponse B : 49
        - Réponse C : 33
        - Réponse D : 3A
- Personnage n°3 :
    - Nom : Scarlett
    - Position dans le labyrinthe :
    - Particularité : Ce personnage propose 4 objets différents, une ventouse WC, un frisbee, une épée ou une batterie externe, le joueur doit choisir l'objet qu'il veut garder pour affronter un monstre se situant exclusivement dans le bon couloir.

En fonction de ce que choisit le joueur, la situation sera différente et burlesque, il n'y aura pas de véritable affrontement.

- Personnage n°4 :
    - Nom : Raymond Poulidor
    - Position dans le labyrinthe :
    - Particularité : C'est un cycliste, il a donc son vélo avec lui.

    - Énigme : "Je monte une pente à 12km/h. Puis vient la descente qui fait la même distance que la montée. A quelle vitesse je devrais descendre pour avoir une vitesse moyenne sur la montée et la descente de 24 km/h ?"
        - Réponse A : 60 km/h
        - Réponse B : 72 km/h
        - Réponse C : 68 km/h
        - Réponse D : 52 km/h

- Personnage n°5 :

    - Nom : Timmy
    - Position dans le labyrinthe : *FIN*
    - Particularité : Il s'agit du petit garçon ayant forcé le personnage à jouer. Il commence par annoncer qu'il n'y avait aucune bonne réponse à l'énigme précédente.

Contraintes techniques

Le projet est basé sur le principe d'*ASCII art*, en effet le programme affichera seulement des symboles présents dans la table ASCII directement dans le terminal UNIX de l'utilisateur.

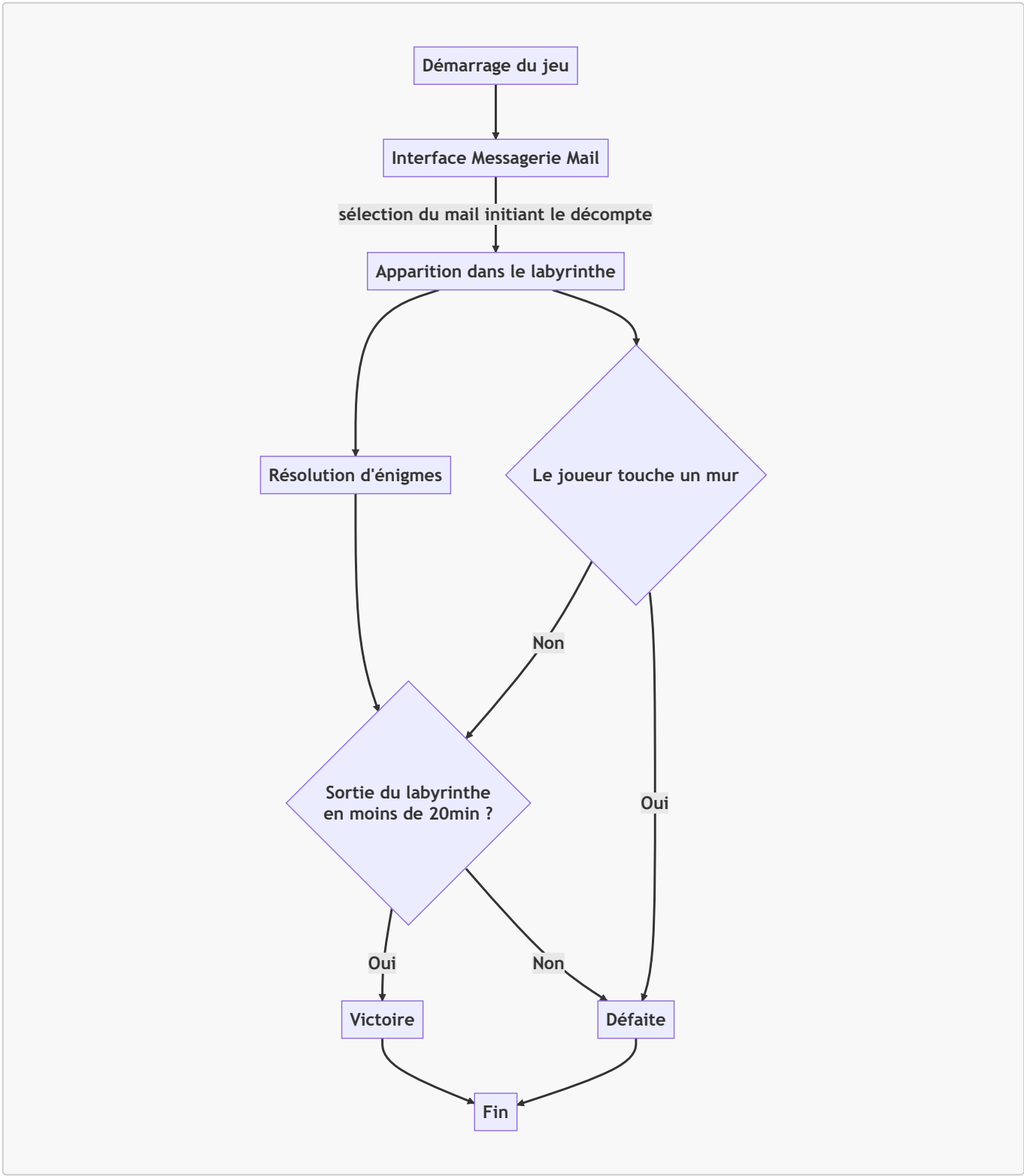
ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

A noter qu'au vu de la définition du jeu (le terminal est dézoomé), le texte plein ne devra pas être affiché directement sans traitement, l'utilisation d'un convertisseur en ASCII-ART est **nécessaire**.

Le jeu sera codé en Python. Il utilise - dans sa première version - exclusivement le paradigme impératif, avant d'espérer une migration complète ou partielle vers le paradigme de programmation orientée objet.

Scenario d'utilisation



Analyse du besoin

Fonctionnalités

Le joueur doit pouvoir se déplacer librement dans l'espace que représente le labyrinthe, pour cela les images générées par le moteur graphique doivent être véridiques en toute position et orientation dans l'espace de jeu. De même, la position exacte du joueur sera constamment recalculée pour s'assurer qu'il se situe en permanence dans une zone jouable.

Une utilisation ludique des objets ainsi qu'une manipulation logique des interfaces est nécessaire. Comme il est inscrit au niveau du règlement, la réalisation d'un compteur numérique de couleur rouge est à mettre en place. Le joueur doit toujours avoir le choix entre les différents chemins à emprunter.

Des transitions entre les différentes interfaces composant le jeu sont à implémenter.

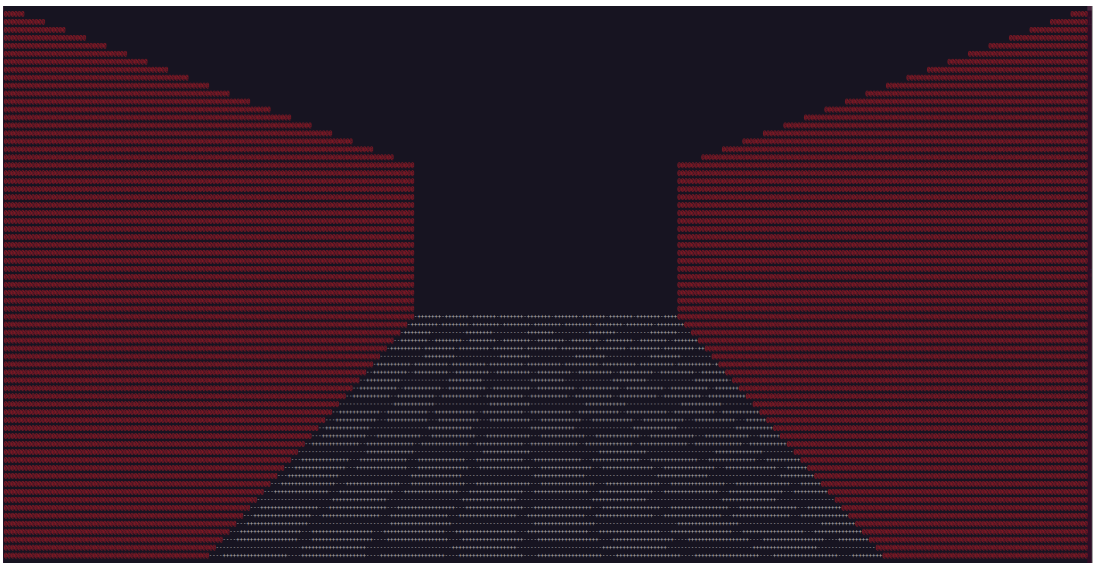
Prototypage

La toute première version consistait en une image dynamique de couloir, le projet a ensuite migré vers un système automatisé, cependant cela permet d'appréhender pleinement la qualité de rendu envisageable.

Pour simuler un effet de profondeur 3D, je joue sur la perspective du rendu. Un moyen simple est de générer un couloir dont le sol et les murs se rapprochent du centre de l'image (fixé à une certaine profondeur de champ). Cette profondeur a été choisie arbitrairement et je réfléchis à la faire varier en fonction de futures contraintes inhérentes au *gameplay*.

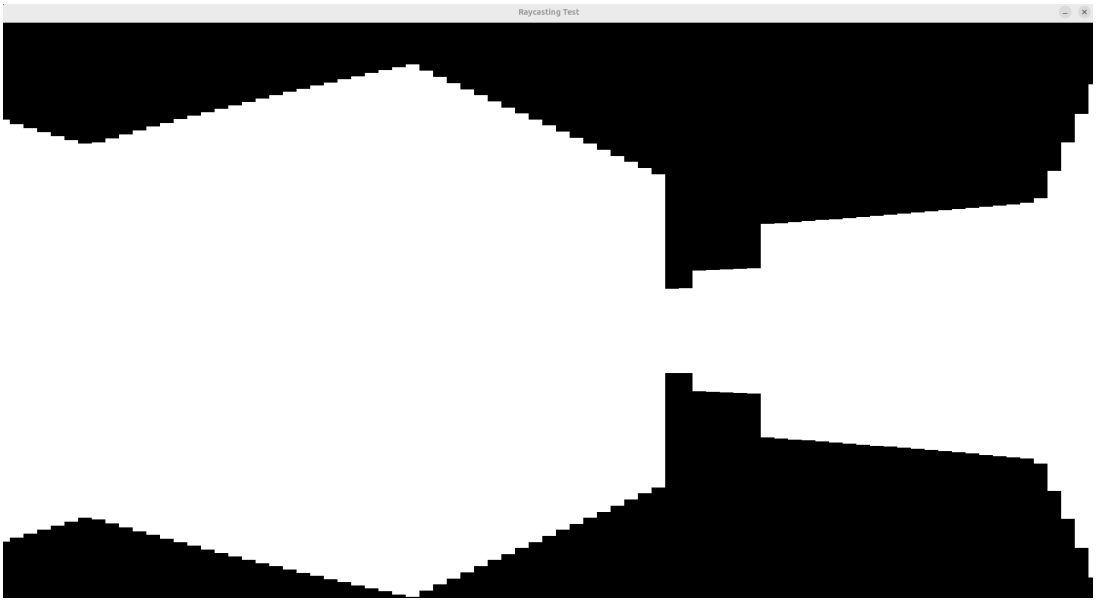
Pour donner à l'utilisateur l'impression d'avancer ou de reculer, l'algorithme génère continuellement une texture pavée au sol, les interstices entre les différentes pierres reculent dès lors que le joueur avance, et inversement ils avancent dès lors que le joueur recule. Pour accentuer cette sensation, je considère l'idée d'ajouter des repères fixes sur les murs, tels que des fenêtres ou des tableaux.

En choisissant des paramètres proportionnels, on obtient finalement le résultat suivant :



Le projet complet est en accès libre pour toute consultation/exécution/modification à cette adresse : **cliquez ici**. L'ensemble de mes avancées sur l'algorithme y seront notamment visibles.

Un deuxième prototype avait pour objectif de démontrer le bon fonctionnement de mon système de raycasting, avant de se lancer dans le rendu total en ASCII. Pour cela j'ai d'abord développé une méthode de rendu géométrique au moyen des outils proposés par la librairie *PyGame*. J'en suis arrivé au résultat suivant :



Livrables

- **Cahier des charges** (*ce document*). Plusieurs versions sont disponibles dans le gestionnaire :

Date	Version	Téléchargement
10/02/2025	`v0.1`	**Télécharger**
21/02/2025	`v0.2`	**Télécharger**

- **Conception**, maintenu à cette adresse : *Document indisponible*
- **Codage Procédural**, maintenu à cette adresse : *Document indisponible*
- **Codage Orienté Objet**, maintenu à cette adresse : *Document indisponible*