

Rapport SI28

écriture interactive et multimédia

Mission évasion



Julien Hellec - Chloé Arhant - Chloé Taurel - Nouria-Louise Idrissa

Sommaire :

Note d'intention :	2
Concept :	2
Public :	2
Objectif :	2
Cahier des charges :	3
I. Généralités :	3
II. Ressources Médias :	4
III. Structure et navigation :	4
II. Formes et degrés d'interactivité :	4
III. Choix techniques :	5
Scénario:	9
I. Les énigmes	9
II. Les différentes fins	13
1. Mort par explosion	13
a) Par Time-Out	13
b) Par Bouton rouge	13
2. Mort par asphyxie	13
3. Retour sur Terre	13

Note d'intention :

Concept :

Le concept de ce projet est de jouer à un jeu interactif de type escape game. En effet, le joueur entre dans la peau d'un astronaute de l'ISS qui doit à tout prix trouver un moyen de rentrer sur terre avant que l'ISS n'explose. En résolvant des énigmes et en collectant des indices, le joueur va chercher à s'infiltrer dans la fusée de secours lui permettant de survivre.

Le joueur évolue dans un environnement spatial avec la possibilité de découvrir plusieurs pièces de l'ISS, et d'interagir avec des objets.

En ce qui concerne le scénario de cet Escape Game, nous avons choisi un sujet d'actualité comme base. Il porte sur des astronautes en mission dans l'ISS qui, à cause d'un dysfonctionnement de leur module d'envoi, se retrouvent bloqués jusqu'en février 2025 dans l'espace. L'ironie du sort fait que c'est Space X (à savoir une entreprise concurrente à la leur) qui est chargée de les ramener sur Terre.

Dans notre version modifiée, Space X refuse d'accepter l'astronaute dans sa fusée sous prétexte que leurs entreprises sont concurrentes. Seulement l'ISS n'a plus que peu de temps devant elle, et le refus de Space X condamne notre astronaute à s'éteindre avec la station.

Public :

Ce projet vise un public large. En effet, tant que le joueur est curieux, aime résoudre des énigmes ou tout simplement jouer à des jeux, il pourrait être intéressé par cet Escape Game.

Pour que le public visé soit large, il est nécessaire que les énigmes soient variées et de niveau modéré. Tous les indices nécessaires à la résolution du jeu sont à la disposition de l'utilisateur, afin qu'il puisse être terminé sans avoir besoin de connaissances extérieures.

Objectif :

A travers ce projet, nous souhaitons faire transparaître le stress et l'injustice que pourrait ressentir un astronaute dans cette situation. Le format de l'Escape Game est on ne peut plus adapté pour transmettre ce genre d'émotions. Le temps limité et le chronomètre qui défile accentuent la pression en ajoutant un ultimatum au joueur, tandis que les énigmes peuvent amener le joueur à se sentir impuissant s'il se retrouve dans une impasse.

Nous avons souhaité faire une critique du capitalisme et du nationalisme grandissant, notamment aux Etats-Unis, grâce à notre sujet d'actualité modifié, montrant le manque d'humanisme dont certains haut-décisionnaires font preuve lorsque celle-ci implique une perte de bien.

Cahier des charges :

I. Généralités :

- Rester fidèle à la l'ISS :

Nous souhaitons retranscrire au maximum la disposition ainsi que les modules de l'ISS dans leur spatialité. Nous avons donc choisi d'utiliser des captures 3D de l'ISS ainsi que de construire nos énigmes autour des éléments présents dans les différents modules que nous explorons.

- Le joueur doit se sentir sous pression :

Le type même de notre projet requiert une certaine pression à appliquer sur le joueur. De plus, il semble essentiel que le joueur ressente la pression qu'aurait ressenti un astronaute en situation réelle.

- Des possibilités de fin multiples :

Au sein de l'ISS, il existe d'innombrable manière de mourir, nous souhaitons retranscrire ce danger permanent à l'utilisateur. Nous allons donc ajouter des fins alternatives menant à la mort du joueur.

- Une exploration gratifiante :

Pour respecter le genre de l'escape game, il est essentiel que le joueur puisse interagir avec différents indices qu'ils soient réels ou factice pour obtenir des indices quand à la résolution des différentes énigmes de l'Escape Game.

- Tenter de recréer un Storygame :

Dans un Storygame, les choix du joueur ont un impact sur l'histoire et le déroulement du jeu. Nous souhaitons tenter de reproduire ce genre pour que l'expérience de chaque joueur soit unique.

II. Ressources Médias :

Nous avons choisi d'utiliser pour l'aspect graphique de notre Escape Game les images de la visite 3D de l'ISS disponible sur le site de la NASA, au lien suivant :

<https://www.nasa.gov/feature/iss-virtual-tour/>

Nous avons fait ce choix afin d'avoir un graphisme cohérent dans le temps au cours du jeu, mais également afin d'être cohérents avec la structure réelle de l'ISS. Initialement, nous souhaitions utiliser l'IA pour créer des images plus esthétiques, mais cela s'est révélé impossible car toutes les IA que nous avons testées produisaient des images incohérentes avec le manque de gravité dans l'espace, ou nous produisaient des images dignes d'un vaisseau de la Guerre des Étoiles. Ce choix final s'est donc avéré beaucoup plus cohérent, en plus d'être beaucoup plus simple.

III. Structure et navigation :

La navigation au sein de l'ISS est assez simple. Nous avons positionné des boutons latéraux sur la totalité de la hauteur de l'écran pour permettre au joueur de se déplacer autour de lui lorsque cela est possible. Il est également possible pour le joueur de se déplacer dans les différentes pièces de la station en cliquant sur les différentes portes accessibles. Enfin, il lui est possible de revenir dans la pièce précédente en utilisant les boutons situés sur le bas de l'écran.

Nous avons choisi cette solution pour plusieurs raisons. Tout d'abord elle était simple à implémenter mais également car c'est la solution qui nous a paru la plus simple d'utilisation pour l'utilisateur. En effet, cela revient globalement à se déplacer à l'aide de flèche mais les boutons étant transparents, cette solution se révèle plus esthétique.

IV. Formes et degrés d'interactivité :

Il y a 2 formes d'interactivité majeures dans notre projet, la navigation et l'interaction avec le décor. Il est important de noter que si le joueur n'interagit pas avec le jeu, rien ne se passe, le jeu ne démarre même pas. L'interactivité est donc une variable essentielle à notre projet et sans laquelle il n'a pas grand intérêt.

Concernant la navigation, le joueur est maître de ses mouvements, il se déplace librement dans la station dans le temps imparti. Libre à lui de se déplacer pour trouver toute forme d'indice ou alors de tenter de résoudre les énigmes par la force brute en testant toutes les combinaisons possibles.

C'est justement dans cette phase de recherche d'indices que l'interactivité avec le décor rentre en jeu, cliquer sur les boutons présent à l'écran lui permet soit de trouver les énigmes soit les indices qui les mèneront jusqu'à l'énigme ou encore les moyens qui lui permettront de la résoudre. Nous avons tenté de varier les interactions, même si elles sont majoritairement liées à l'appui de bouton dans l'air de jeu. L'audio est une composante essentielle de notre projet, nous pouvons par exemple noter l'indice audio donnée par la radio lorsque celle-ci est correctement paramétrée. De plus, la musique d'ambiance et l'alarme sont importantes pour que le joueur se sente oppressé par l'urgence de la situation dans lequel il se trouve.

En effet, nous avons fait plusieurs choix pour alimenter l'ambiance angoissante que nous souhaitons. Comme mentionné, la musique de fond et l'alarme qui restent tout le long de la partie en sont les composantes auditives. Nous avons également choisi de mettre un Timer pour respecter le principe de l'échappement game. En plus de ce timer, nous avons ajouté un filtre rouge qui se répartit sur toute l'image, dont l'intensité et la fréquence augmentent avec le temps. Nous avons choisi d'ajouter cet aspect pour que le Timer ne soit pas la seule notion du temps que le joueur puisse avoir. Grâce au filtre rouge, plus le temps s'écoule, plus le joueur voit fréquemment cette alarme, qui ajoute à la sensation d'urgence d'évacuer.

L'utilisateur a exactement 10 minutes pour évacuer, c'est un temps très court compte tenu du nombre d'informations qu'il faut réunir. Nous avons fait exprès de rendre ce temps assez court pour que l'utilisateur soit éventuellement obligé de recommencer plusieurs fois.

V. Choix techniques :

Le premier choix technique que nous avons dû faire correspondait à la plateforme sur laquelle nous allions choisir de développer notre échappement game. Il nous est vite apparu que la solution optimale était un jeu dans le WEB. Deux solutions techniques se sont alors offertes à nous. La première était de coder l'Escape Game depuis une page blanche à l'aide d'HTML, CSS et Javascript par nous même. Cette solution aurait pu être viable si tous les membres du groupe étaient familiers avec ce genre de technologie car elle est plus difficile à prendre en main. La seconde, était d'utiliser le logiciel Twine, qui permet à partir d'une syntaxe propre (ici SugarCube) de générer du code HTML, CSS et Javascript automatique une fois l'export fait. Nous avons décidé de retenir cette solution, car à première vue il semble évident que pour un groupe composé à moitié d'étudiants non GI il est plus facile d'utiliser une syntaxe simple que d'apprendre 3 langages pouvant se révéler complexes.

Toutefois, pour des étudiants GI ayant l'habitude de traiter avec différents langages, la documentation Twine SugarCube s'est révélée particulièrement incomplète et même par moment obsolète (voire fausse :(). En plus du manque de documentation sur SugarCube,

Twine n'étant que très peu utilisé par la communauté, il n'y a pas non plus beaucoup de forum traitant des différents problèmes que l'on peut rencontrer.

Il faut tout de même souligner que Twine SugarCube à quelques bons côtés. Il est simple d'accès mais également très largement suffisant pour des projets qui cherchent juste à créer de l'exploration sans trop d'interactions complexes (malheureusement, notre projet en est tout le contraire). Il permet également de créer des liens automatiques entre les différentes pages WEB de notre projet, c'est un gain de temps majeur.

Nous avons par la suite créé un cadre pouvant être copié dans tous les autres passages du projet. Ce cadre comprend des boutons de navigation basique déjà dimensionnés et positionnés ainsi que le container permettant d'accueillir l'affichage du projet. Créer ce cadre à permis aux étudiants non GI du groupe de pouvoir également participer au code du projet, car il leur suffisait de copier coller cette partie et simplement changer ce qui les intéressait. En effet, les container permettait d'appliquer un code CSS fait à l'avance permettant de calibrer l'affichage du projet.

Parmi les autres solutions techniques intéressantes que nous avons employées, nous pouvons noter une machine à état pour la réalisation du digicode. Nous avons initialement souhaité faire une fonction javascript permettant la comparaison entre la chaîne de caractère entrée dans le TextEntry et le code choisi. Cependant, il s'avère qu'il nous a été impossible de lire et modifier une variable SugarCube dans la partie Javascript du projet. Nous avons donc trouvé la solution d'une machine à état.

Cette solution est habituellement utilisée dans les systèmes électroniques, elle permet de faire différentes actions en fonction de l'état dans lequel se trouve le système. Dans notre cas, les états sont symbolisé par les passages suivants :

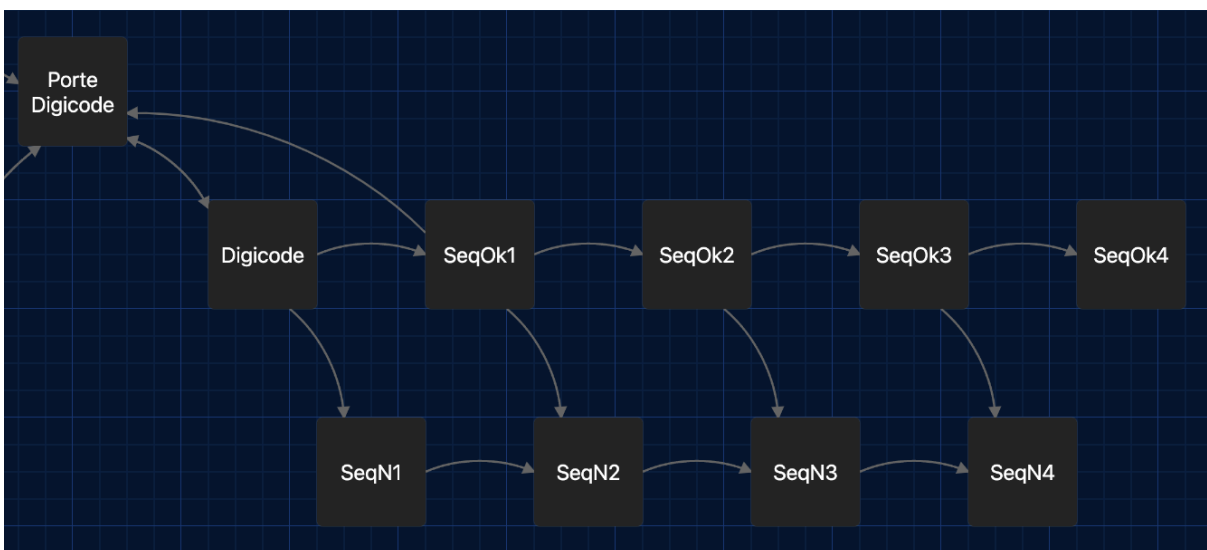


Figure 1 : Machine à états du digicode

Ici, le passage à l'état "SeqOk[x]" correspond à la pression du bon numéro dans le code, tandis que le passage à l'état "SeqN[x]" correspond à la pression du mauvais numéro dans la séquence. Ainsi, nous détectons via l'arrivée dans le passage "SeqOk4" que le code entrée est correct et nous donnons accès à la suite du jeu au joueur. Alors que dans le cas où le joueur commet une erreur dans le code, nous jouons un bruit d'erreur à l'arrivée dans le passage "SeqN4" et nous l'envoyons au passage Digicode.

Chaque passage ayant le même visuel, l'utilisateur ne se rend pas compte des différents passages. Ils nous permettent simplement de compter le nombre de chiffres entrées ainsi que de la validité du code.

Pour plus de lisibilité sur le graphe des passages généré par twine, nous avons fait le choix de ne pas afficher les retours vers la Porte au milieu d'une séquence, qui sont néanmoins possibles (via la commande *onclick="SugarCube.Engine.play('Passage')"* qui permet d'aller dans un Passage en appuyant sur le bouton de retour, sans afficher la flèche sur le diagramme). Les différentes possibilités ne sont donc pas toutes visibles sur la Figure 1, nous avons seulement souhaité y représenter la machine à état.

Nous avons également employé le même principe pour l'orientation du bouton des degrés lors de la sélection de la fréquence.

La technique employée pour la vérification du mot de passe du fichier confidentiel dans l'ordinateur de la station est totalement différente. Etant donné que dans ce cas le joueur entre la totalité de la chaîne de caractère avant d'appuyer sur le bouton de validation, nous avons simplement pu utiliser une fonction Twine permettant de comparer la chaîne de caractère avec le code souhaité.

Si le joueur ne rentre pas le bon mot de passe, nous le redirigeons vers un passage lui affichant un message d'erreur tandis que si le code est correct, nous le redirigeons vers le passage d'accès au fichier.

Un autre des défis de notre projet à été la conception du Timer. En théorie, cela n'aurait dû être qu'une formalité. Cependant, l'impossibilité de modifier des variables Twine depuis le Javascript nous a empêché d'utiliser ce langage pour le Timer. La solution que nous avons choisie est donc plus précaire et pose quelques problèmes qui seront mentionnés plus tard dans ce rapport.

L'implémentation du Timer est donc directement faite depuis Twine à l'aide d'un passage Timer que nous ajoutons à chaque passage à l'aide de l'instruction :

```
<<include "Timer">>
```


Celle-ci permet d'inclure le Timer dans tous les passages dès lors que le jeu commence. Le fonctionnement du Timer en lui-même est assez simple. Il comprend une div pour positionner et modifier le style du Timer ainsi qu'un algorithme qui décompte les minutes et secondes jusqu'à 0. Une fois arrivée à 0 le Timer envoie le joueur dans le passage de fin.

Comme nous l'avons dit plus tôt cette méthode pose des problèmes comme par exemple, le fait qu'il est très compliqué d'afficher devant les secondes un 0 lorsqu'elles sont inférieures à 10 ou encore que le délais entre chaque seconde est variable car la fonction `<<repeat 1s>>` n'est pas très fiable. Un autre souci rencontré à cause de cette implémentation de Timer particulière est que le temps affiché ne s'écoule pas indépendamment des changements de passage. En effet, au moment où l'utilisateur change de passage, les variables du Timer ne changent pas. Ainsi, si l'utilisateur clique très vite sur les boutons de navigation, le temps ne s'écoule pas. Nous avons décidé de faire abstraction de ce problème, d'une part par manque d'autre solution, et d'autre part car si l'utilisateur clique frénétiquement sur des boutons il n'aura pas le temps de comprendre les énigmes du jeu, donc le Timer n'avancerait pas, mais le joueur non plus.

Scénario:

L'utilisateur commence l'aventure dans la zone Zvezda de l'ISS (voir *Figure 2* ci-dessous). Son but est d'atteindre la fusée amarrée au node 2 et ainsi se sauver de la station spatiale qui est sur le point d'exploser. Pour cela, il devra résoudre plusieurs énigmes tout au long de son parcours et ainsi atteindre la fusée. Tout au long de la navigation, quelques indices seront placés pour faire comprendre à l'utilisateur ce qu'il doit faire.

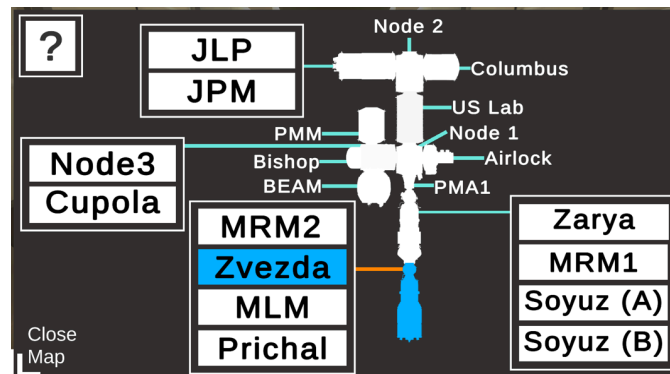


Figure 2 : Plan de l'ISS

Première image:

L'introduction de notre escape game est sous forme de vidéo. Celle-ci présente une série de premières pages de journaux, qui défilent et se superposent à l'écran. Chaque journal est accompagné d'extraits audio des plateaux télé parlant de cette nouvelle. Les news diffusées reprennent les événements que nous avons évoqués lors de la présentation du projet : une mission spatiale sur l'ISS où, à cause d'un dysfonctionnement du module de communication, les astronautes se retrouvent coincés dans l'espace.

Une fois cette introduction terminée, un texte s'affiche et décrit la situation de l'utilisateur. Il se retrouve ensuite directement face à des ordinateurs au cœur de l'ISS.

I. Les énigmes

Énigme 1 : Dossier confidentiel

Avant de pouvoir commencer la navigation au travers de l'ISS, il faudra que l'utilisateur clique sur l'ordinateur, qui contient de nombreuses informations cruciales sur le déroulement des énigmes. Le joueur peut trouver dans l'ordinateur un plan qui explique la fonction de chaque module. Il pourra ainsi comprendre où est la fusée et où il doit se déplacer. Pour cela, il faudra débloquent le dossier technique avec un code caché dans

l'ordinateur. L'utilisateur va alors essayer de se balader à travers l'ordinateur pour trouver le code. Un mail est disponible expliquant que mettre une date pour protéger les dossiers de l'ISS n'est pas une bonne idée. Ainsi, l'utilisateur sait qu'il doit rechercher une date pour déverrouiller le dossier. Cette date se trouve en fouillant l'ordinateur, au sein de l'onglet "Paramètres". Pour que cette énigme ne soit pas trop difficile à résoudre, nous avons délibérément mis très peu d'éléments sur l'ordinateur. Par exemple, il n'y a qu'un seul mail dans la boîte mail, qui est celui mentionné précédemment. L'utilisateur ne peut visiter que la boîte mail, les documents et images qui ne contiennent pas beaucoup d'éléments.

Une fois le dossier confidentiel débloqué, l'utilisateur a accès à la description des modules et au dossier technique. Grâce au premier document, l'utilisateur sait désormais que les vaisseaux SpaceX peuvent amarrer au niveau de deux modules : Node 2 ou PMA. Il peut mettre en relation ces informations avec le plan de l'ISS donné dans les images de l'ordinateur afin de savoir où se trouvent les modules dans la station. Grâce au dossier technique, l'utilisateur sait qu'un sas ne doit pas être ouvert sans l'avoir correctement pressurisé. Enfin, dans le dossier confidentiel l'utilisateur a accès à un descriptif du fonctionnement de la pressurisation des SAS. C'est grâce à ce document qu'il sait comment dépressuriser le SAS de Node 2 avant de l'ouvrir.

Solution : le code à entrer est "201198".

Énigme 2: Pressurisation

Si l'utilisateur a bien ouvert le dossier confidentiel, il a toutes les cartes en main pour pouvoir dépressuriser le SAS de Node2 avant de s'attaquer à l'énigme 3 (qui consiste à ouvrir le SAS pour accéder à la fusée). Cette étape est cruciale, car comme mentionné dans le dossier technique, l'utilisateur doit impérativement ajuster la pression d'un SAS avant de l'ouvrir, sans quoi il mourra d'asphyxie.

Dans le dossier confidentiel, il se trouve un fichier guidant la dépressurisation dans lequel il est mentionné que la dépressurisation des SAS se fait dans le module Airlock, sur l'un des trois digicodes. Il est également précisé que pour ouvrir le module Node2, il faut utiliser le digicode de droite. Enfin, il est indiqué que les codes à entrer se trouvent dans le SAS Bishop. À l'aide du plan, le joueur peut se rendre dans le SAS Bishop pour trouver cette liste, et ainsi rentrer le code sur le bon digicode du module Airlock afin de dépressuriser le SAS Node 2, **avant de l'ouvrir**. Si l'utilisateur entre le bon code sur la porte Node 2 avant de dépressuriser le SAS, il mourra d'asphyxie. Dans le document, d'autres SAS sont mentionnés mais nous n'en avons pas implémenté l'existence, ils sont seulement mentionnés pour que l'utilisateur aie à savoir qu'il doit ouvrir la porte Node 2 et donc utiliser le digicode de droite.

Une fois le bon code entré, un audio de dépressurisation sera lancé, informant l'utilisateur qu'il a bien pressurisé le SAS, et qu'il peut donc passer à l'énigme de son ouverture (voir *énigme 3 : la radio*).

Solution : le code "3-8-6" doit être entré dans le digicode de droite pour dépressuriser Node 2.

Enigme 3 : la radio

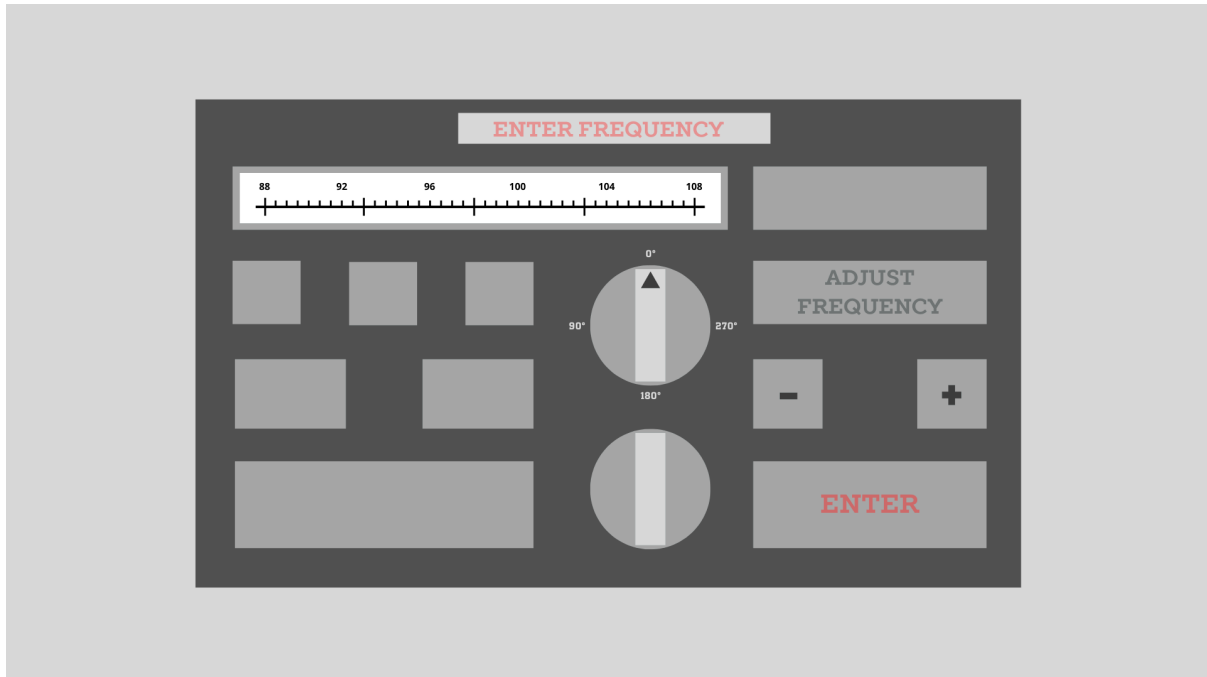


Figure 3 : Image de la radio

Le joueur doit trouver la bonne fréquence et le bon angle de l'antenne afin de découvrir le message brouillé par la radio. Ce message permet de connaître le code du sas permettant de rejoindre la fusée.

Afin de trouver la bonne fréquence et le bon angle, le joueur va devoir se servir de l'ordinateur positionné à côté de la radio. Sur cet ordinateur, on découvre un blue_screen (écran de bug) sur lequel on peut lire la fréquence et l'angle en binaire. Grâce aux fiches positionnées sur l'autre mur, le joueur va devoir faire la conversion et retrouver les chiffres correspondants.

```

*** SYSTÈME_DÉCONNECTÉ -- IMPOSSIBLE_DE_RÉTABLIR_LA_CONNEXION --

Le terminal principal est hors ligne - toutes les opérations sont suspendues
Pour continuer il est impératif de rétablir une synchronisation manuelle

Pour accéder au sas de sortie vous devez ajuster les paramètres sur la radio
La fréquence de synchronisation doit être réglée sur [01101001] Hz

Les interférences extérieures pourraient perturber le signal si les réglages sont incorrects
L'antenne de communication doit être orientée à un angle précis de [0x10E] degrés

Toute déviation des paramètres indiqués annulera la procédure

L'accès au code d'ouverture du sas ne sera possible qu'après synchronisation complète
Les erreurs persistantes ne seront pas corrigées automatiquement

*** FIN_DU_MESSAGE_SYSTÈME --
  
```

Figure 4 : indices fréquences et polarisation

CHIFFRE	HEXADECIMAL	NOMBRE	BINAIRE
0	0x000	88	01011000
45	0x02D	89	01011001
90	0x05A	90	01011010
135	0x087	91	01011011
180	0x0B4	92	01011100
225	0x0E1	93	01011101
270	0x10E	94	01011110
315	0x13B	95	01011111
360	0x168	96	01100000
		97	01100001
		98	01100010
		99	01100011
		100	01100100
		101	01100101
		102	01100110
		103	01100111
		104	01101000
		105	01101001
		106	01101010
		107	01101011
		108	01101100

Figure 5 : fiches explicatives du binaire et hexadécimal

De plus, il va se servir de la fiche utilisation de la radio qu'il peut apercevoir à côté de celle-ci. Grâce à cette fiche, il va comprendre comment et pourquoi aligner la bonne fréquence et le bon angle.

Solution : il faut rentrer la fréquence 105 Hz et la polarisation 270° pour entendre le code du SAS.

Énigme 4 : le sas Node 2

Cette dernière énigme est évidemment entremêlée à toutes les énigmes précédentes : il faut que l'utilisateur aie compris grâce à l'ordinateur qu'il doit ouvrir le SAS de Node2, qu'il l'ait bien dépressurisé préalablement, et qu'il ait trouvé le code en réussissant à trouver la bonne fréquence radio (le code est un audio répété en boucle, une fois que les bons paramètres ont été entrés sur la radio). Une fois la porte ouverte, l'utilisateur n'a plus qu'à cliquer sur la dernière porte, celle de Node2, pour entrer dans la fusée SpaceX et revenir sur Terre.

Solution : le code à entrer sur le digicode de la porte Node 2 est "7424"

II. Les différentes fins

1. Mort par explosion

a) Par Time-Out

Une fois le temps écoulé, c'est-à-dire une fois que le timer rouge arrive à 0, la station arrive en fin de vie et active son auto-destruction. Ainsi, comme dans tout bon escape-game qui se respecte, le jeu se termine si l'utilisateur n'a pas pu s'échapper à temps.

b) Par Bouton rouge

La station peut également exploser d'une autre manière : si l'utilisateur clique sur un gros bouton rouge, présent sur la gauche de la pièce de départ. Ce bouton est un bouton à ne surtout pas activer car il enclenche immédiatement l'autodestruction de la station, et un astronaute qui ne sait pas qu'il ne faut pas y toucher est très mauvais dans son métier...

2. Mort par asphyxie

Comme mentionné à plusieurs reprises dans ce rapport, l'utilisateur doit absolument dépressuriser le SAS Node 2 avant de l'ouvrir. Si par malheur il ouvre la porte en entrant le bon digicode avant d'avoir effectué cette tâche, il se verra mourir d'asphyxie et le jeu sera terminé.

3. Retour sur Terre

Il existe évidemment une fin heureuse pour le joueur. S'il a résolu toutes les énigmes et a réussi à atteindre la fusée avant la fin du Timer, il pourra alors revenir sur Terre et l'Escape Game est gagné !