|  |  |  |  |
| --- | --- | --- | --- |
| **Analyse mini projet : Mots Cachés**  **Document Technique** | | | |
| **Document Technique** | | | |
| Référence | YNOV-2019-03-DTH-001 | Auteurs | D.Palermo |

|  |  |  |
| --- | --- | --- |
|  | **Rédacteur** | **Vérificateur** |
| **Nom** |  |  |
| **Date** |  |  |
| **Visa** |  |  |

|  |  |  |
| --- | --- | --- |
| **Diffusion** | **Statut** | **Nom** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Historique des modifications | | | |
| **Version** | **Pages** | **Description de la modification - Auteurs** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**SOMMAIRE**

[1 Présentation 4](#_Toc3302347)

[1.1 Introduction 4](#_Toc3302348)

[1.2 Glossaire 4](#_Toc3302349)

[1.2.1 Abréviations & Glossaire 4](#_Toc3302350)

[1.2.2 Documents applicables 4](#_Toc3302351)

[1.2.3 Documents de référence 4](#_Toc3302352)

[2 Mini-projet : Mots cachés 5](#_Toc3302353)

[2.1 besoins fonctionnels 5](#_Toc3302354)

[2.1.1 Utilisateurs du projet SCONE 5](#_Toc3302355)

[2.1.2 Enchaînement 6](#_Toc3302356)

[2.1.3 Analyse métier statique 10](#_Toc3302357)

[3 ANNEXES 11](#_Toc3302358)

[3.1 ANNEXE : Sujet mini projet 11](#_Toc3302359)

[3.2 ANNEXE : Notation UML (résumé) 12](#_Toc3302360)

[3.3 ANNEXE : Bibliographie technique 19](#_Toc3302361)

**TABLEAUX**

[Tableau 1‑1 : Abréviations & Glossaire 4](#_Toc3302362)

[Tableau 1‑2 : Documents applicables 4](#_Toc3302363)

[Tableau 1‑3 : Documents de référence 4](#_Toc3302364)

[Tableau 3‑1 : Notation UML : Diagramme de cas d'utilisation 13](#_Toc3302365)

[Tableau 3‑2 : Notation UML : Diagramme de classes 15](#_Toc3302366)

[Tableau 3‑3 : Notation UML : Diagramme de séquence 17](#_Toc3302367)

[Tableau 3‑4 : Notation UML : Diagramme de package 18](#_Toc3302368)

[Tableau 3‑5 : Bibliographie Technique 19](#_Toc3302369)

**FIGURES**

[Figure 2‑1 : la vue du joueur 5](#_Toc3302370)

[Figure 2‑2 : Mini Projet – Enchaînement général 6](#_Toc3302371)

[Figure 2‑3 : Générer la grille de mots cachés 7](#_Toc3302372)

[Figure 2‑4 : Jouer 8](#_Toc3302373)

[Figure 2‑5 : Jouer : partie Jouer sur ordinateur 9](#_Toc3302374)

[Figure 2‑6 : Jouer : partie Jouer format papier 9](#_Toc3302375)

[Figure 2‑7 : package MotsCaches 10](#_Toc3302376)

# Présentation

## Introduction

Ce document technique décrit l'analyse orientée objet du mini projet de génération de Mots cachés.

Les annexes contiennent :

1. Sujet mini projet
2. Une présentation résumée de la notation UML utilisée dans le document.
3. Le code couleur UML utilisé dans le document.
4. Une bibliographie technique sur UML.

## Glossaire

### Abréviations & Glossaire

|  |  |
| --- | --- |
| **Abréviation** | **Signification** |
| **UML** | **U**nified **M**odelling **L**anguage |

Tableau ‑ : Abréviations & Glossaire

### Documents applicables

|  |  |
| --- | --- |
| **Identification** | **Description** |
| [REF ] | Support de cours : Yantra-Technologies-UML2-V3.2Light.pdf de D.Palermo |

Tableau ‑ : Documents applicables

### Documents de référence

|  |  |
| --- | --- |
| **Identification** | **Description** |
| [APP ] | Sujet Mini Projet langage C : D.Palermo ( fourni en ANNEXE : Sujet mini projet Sujet mini projet ) |

Tableau ‑ : Documents de référence

# Mini-projet : Mots cachés

Le but de ce mini-projet est de créer un générateur en langage C de grille aléatoire de mots cachés au format texte ( minimum).

## besoins fonctionnels

Ce paragraphe va décrire les besoins fonctionnels du mini-projet

### Utilisateurs du projet SCONE

Le mini-projet sera utilisé par un seul type d'acteur et qui sera nommer le joueur .

Le joueur devra pouvoir :

* Générer la grille de mots cachés
* Jouer de 2 manières différentes :
  + Jouer via une grille imprimer
  + Jouer directement sur l'ordinateur



Figure ‑ : la vue du joueur

### Enchaînement

Le logiciel qui va permettre de générer la grille de mots cachés est fait de 2 enchaînent principaux :

* La génération de grille de mots cachés
* la partie qui permet de jouer avec la grille générée



Figure ‑ : Mini Projet – Enchaînement général

#### Traitement : Générer la grille de mots cachés



Figure ‑ : Générer la grille de mots cachés

Le joueur fournit la taille de la grille a généré.

Si la grille peut être générer ( taille vérifier) alors l'utilisateur demande la génération de la grille

#### Traitement : Jouer



Figure ‑ : Jouer

Le joueur peut demander une fois la grille générer soit :

* D'imprimer la version papier de la grille ( voir Figure 2‑6 : Jouer : partie Jouer format papier )
* De jouer directement sur l'ordinateur ( voir Figure 2‑5 : Jouer : partie Jouer sur ordinateur )



Figure ‑ : Jouer : partie Jouer sur ordinateur



Figure ‑ : Jouer : partie Jouer format papier

### Analyse métier statique

Le logiciel de génération de mot caches sera constitué d'un package MotsCachés. Ce package contient 3 classe

* JeuMotsCaches : classe qui gère le jeu de mots caches :
  + Affichage de la grille
  + Création de la grille
  + Impression de la grille
  + Remplissage de la grille
  + Sauvegarde de la grille
* Grille : classe qui contient la grille des mots cachés
* Dictionnaire : classe qui gère un dictionnaire de mots



Figure ‑ : package MotsCaches

# ANNEXES

## ANNEXE : Sujet mini projet



## ANNEXE : Notation UML (résumé)

|  |  |
| --- | --- |
| **Notion UML** | **Description** |
|  | **Acteur :**  rôle joué par un utilisateur humain ou un autre système qui interagit directement avec le système étudié. Un acteur participe à au moins un cas d’utilisation. |
|  | **Cas d’utilisation (use case)**  Ensemble de séquences d’actions réalisées par le système produisant un résultat observable intéressant pour un acteur particulier. Collection de scénarios reliés par un objectif utilisateur commun. |
|  | **Périmètre du Système**  Il est habituel pour afficher des cas d'utilisation comme étant à l'intérieur du système et les acteurs comme étant en dehors du système. |
|  | **Association**  utilisée dans ce type de diagramme pour relier les acteurs et les cas d’utilisation par une relation qui signifie simplement « participe à ». |
|  | **Extension**  le cas d’utilisation de base en incorpore implicitement un autre, de façon optionnelle, à un endroit spécifié indirectement dans celui qui procède à l’extension |
|  | **Inclusion**  le cas d’utilisation de base en incorpore explicitement un autre, de façon obligatoire, à un endroit spécifié dans ses enchaînements. |
|  | **Généralisation**  les cas d’utilisation descendants héritent de la description de leur parent commun. Chacun d’entre eux peut néanmoins comprendre des relations spécifiques supplémentaires avec d’autres acteurs ou cas d’utilisation.. |

Tableau ‑ : Notation UML : Diagramme de cas d'utilisation

|  |  |
| --- | --- |
| **Notion UML** | **Description** |
|  | **Classe**  description abstraite d’un ensemble d’objets qui partagent les mêmes propriétés et comportements  **Attribut**  donnée déclarée au niveau d’une classe, éventuellement typée, à laquelle chacun des objets de cette classe donne une valeur. Un attribut peut posséder une multiplicité et une valeur initiale. Un attribut dérivé (« / ») est un attribut dont la valeur peut être déduite d’autres informations disponibles dans le modèle.  **Opération ou Méthode:**  élément de comportement des objets, défini de manière globale dans leur classe. Une opération peut déclarer des paramètres ainsi qu’un type de retour. |
|  | **Association**  relation sémantique durable entre deux classes, qui décrit un ensemble de liens entre instances. Une association est bidirectionnelle par défaut, sauf si l’on restreint sa navigabilité en ajoutant une flèche.  **Rôle**  nom donné à une extrémité d’une association ; par extension, manière dont les instances d’une classe voient les instances d’une autre classe au travers d’une association.  **Multiplicité**  le nombre d’objets (min, max) qui peuvent participer à une relation avec un autre objet dans le cadre d’une association. Multiplicités fréquentes :   * + 0..1 = optionnel (mais pas multiple)   + 1 = exactement 1   + 0..\* = \* = quelconque   + 1..\* = au moins 1 |
|  | **Agrégation**  cas particulier d’association non symétrique exprimant une relation de contenance. |
|  | **Composition**  forme forte d’agrégation, dans laquelle les parties ne peuvent appartenir à plusieurs agrégats et où le cycle de vie des parties est subordonné à celui de l’agrégat. |
|  | **Super-classe**  classe générale reliée à d’autres classes plus spécialisées (sous-classes) par une relation de généralisation.  **Généralisation**  relation entre « classifieurs » où les descendants héritent des propriétés de leur parent commun. Ils peuvent néanmoins comprendre chacun des propriétés spécifiques supplémentaires, mais aussi modifier les comportements hérités. |
|  | **Classe d’association**  association promue au rang de classe. Elle possède tout à la fois les caractéristiques d’une association et celles d’une classe et peut donc porter des attributs qui prennent des valeurs pour chaque lien  entre objets. |
|  | **Qualifieur (ou qualificatif)**  attribut qui permet de « partitionner » l’ensemble des objets en relation avec un objet donné dans le cadre d’une association multiple. |
|  | **Dépendance**  relation sémantique entre deux éléments, dans laquelle la modification d’un des éléments peut affecter la sémantique de l’autre élément |

Tableau ‑ : Notation UML : Diagramme de classes

|  |  |
| --- | --- |
| **Notion UML** | **Description** |
|  | **Ligne de vie**  représentation de l’existence d’un élément participant dans un diagramme de séquence. Cela peut être un acteur ou le système en modélisation d’exigences, des objets logiciels en conception préliminaire ou conception détaillée. |
|  | **Message**  élément de communication unidirectionnel entre objets qui déclenche une activité dans l’objet destinataire. La réception d’un message provoque un événement dans l’objet récepteur. La flèche pointillée  représente un retour au sens UML. Cela signifie que le message en question est le résultat direct du message précédent. |
|  | **Occurrence d’interaction**  une interaction peut faire référence explicitement à une autre interaction grâce à un cadre avec le mot-clé ref et indiquant le nom de l’autre interaction.  UML 2 a ajouté une nouvelle notation très utile : les cadres d’interaction. Chaque cadre possède un opérateur et peut être divisé en fragments. Les principaux opérateurs sont :   * loop : boucle. Le fragment peut s’exécuter plusieurs fois, et la condition de garde explicite l’itération. * opt : optionnel. Le fragment ne s’exécute que si la condition fournie est vraie. * alt : fragments alternatifs. Seul le fragment possédant la condition vraie s’exécutera. |

Tableau ‑ : Notation UML : Diagramme de séquence

|  |  |
| --- | --- |
| **Notion UML** | **Description** |
|  | **Package (ou paquetage)**  mécanisme général de regroupement d’éléments tels que classes, interfaces, mais aussi acteurs, cas d’utilisation, etc.  Les packages peuvent être imbriqués dans d’autres packages. |
|  | **Importation**  relation de dépendance entre packages qui rend visibles  les éléments publics de l’un des packages au sein d’un autre |

Tableau ‑ : Notation UML : Diagramme de package

## ANNEXE : Bibliographie technique

|  |  |
| --- | --- |
| **Référence** | **Description** |
|  | UML 2 par la pratique : Etudes de cas et exercices corrigés  Pascal Roques (Auteur) |
| http://ecx.images-amazon.com/images/I/51UrqpfoXjL._SX413_BO1,204,203,200_.jpg | La programmation orientée objet  Huges Bersini (Auteur) |

Tableau ‑ : Bibliographie Technique