# Fonctionnalités

## Fonctionnalités v1

* Affichage Carte
* Hexagones :
  + Bordures noires
  + Bordure intérieure vert clair quand sélectionné
  + Couleur et Image
* Edition d’hexagone
  + Image & Couleur
  + Description
* Chargement / sauvegarde
* Modes :
  + Sélection / Joueur
  + Edition / MJ
    - Edition Carte
    - Edition Hex

## Fonctionnalités v2

* Edition Cours d’Eau & Route
  + Couleur
  + Largeur
  + Suite de point
  + Nom
  + Ajout / Suppression de la ligne

## Fonctionnalités v3

* Redimensionnement carte
* Défilement carte

# Roadmap

## Hexmap

### Données

* Nombre de colonnes (25)
* Nombre de lignes (25)
* Option de comptage des lignes et colonnes ? (0,1, début à gauche, à droite, etc.)

### Fonctionnalités

## Hexagone

### Données

* Coordonnées
* Couleur
* Image
* Description
* Degré d’exploration

### Fonctionnalités

* Sélection (avec liseré vert)

# Divers

\* [X] Lire https://blog.rsuter.com/recommendations-best-practices-implementing-mvvm-xaml-net-applications/

\*\* [X] Mettre en place une ViewModelBase et une INotifyPropertyChanged base classes, qui implémentent les fonctions nécessaires

\*\* [X] Toujours assigner le ViewModel en DataContext

\*\* [X] Ajouter Mytoolkit au projet : https://github.com/RSuter/MyToolkit/blob/master/README.md - le récupérer par NuGet

\*\* [X] Sa logique de commande ==> Menu ?

\*\* [X] Ne pas set le datacontext par binding dans xaml qui instancie la vue; Cf son code ==> méthode perso : get set vm dans le view, view.viewmodel = xxx pour le rajouter

\*\* [X] Explorer sa méthode de comm' pr message, surtout callback ! => à mettre en place d'abord dans le clic sur polygon pour les échanges

\*\* [X] Le choix entre user control et templated control peut attendre d'avoir plusieurs applications

\* [ ] Interface UI : Menu en haut

\*\* [X] Première implémentation

\*\* [X] Menu en haut : récupérer le bouton cliqué , Sinon une fonction différente par bouton

\*\* [X] Menu en haut : command = msg

\*\* [ ] Menu en haut : gérer la structure du menu par un fichier de configuration ? Clairement pas urgent

\* [X] Lien Models <=> Data : on récupère Data, on créée les Models, mais met-on à jour Data quand Model change ? Idéalement, V => VM => M

\* [X] ABSOLUMENT : un système pour gérer toutes ces commandes lancées de partout. On va possiblement devoir disposer de classes supplémentaires pour composer l'application. On a déjà le système des EventMessage d'un côté, EventLogic de l'autre. Il faudrait rassembler les fonctions ainsi : EventMessage les messages, EventLogic les liens EventMessage / GameLogic, et GameLogic les fonctions

\* [X] Sauvegarde / rechargement de Map par Sérialisation

\* [ ] Préparation passage BDD

\*\* [ ] Récupération enum par string (pour BDD)

\*\* [ ] Fichier "constant" pour les couleurs et les images, gestion par ID

\* [X] Faire un MVVM pour la Carte

\* [ ] Faire un MVVM pour l'Interface Gauche (LeftPanel) (au lieu de TileEditor)

\*\* [X] UserControl LeftPanel (simple pour le moment)

\*\* [ ] Menu déroulant contenant les "modes". Edition Hex, Selection Hex, Edition Road/River, etc.

\*\* [ ] Rajouter un champ de texte contenant la Description

\* [ ] Rivières et Routes

\*\* [ ] Réfléchir à comment enregistrer les données de localisation des points d'une rivière. Possiblement s'organiser avec les points centraux comme référence, et des fonctions de traduction

\*\* [ ] River(Data,Model,ViewModel,View)

\*\* [ ] Road(Data,Model,ViewModel,View)

\*\* [ ] Dessiner Rivière, route

\*\* [ ] Sélection Rivière, route

\*\* [ ] Rivière : possibilité d'identifier les hexagones où elle passe ? La direction ?

\* [ ] Au lancement, demander dimensions (et type terrain de base ?)

\* [ ] Regénération d'une carte (avec demande dimensions)

\* [ ] Créer un objet TileSet, contenant la liste des types de terrain, référencé dans Map ?

\* [ ] Ajouter toutes les tiles d'origine dans l'objet TileSet (interface ?), et en faire une version pour chaque set de tiles que j'ai, et faire un Enum + ue fonction pour récupérer le bon

\* [ ] Séparer les fonctions de Save / Load / New dans un autre fichier

\* [ ] Redimensionnement avec boutons +/-

\* [ ] Redimensionnement en cas de redimensionnement de fenêtre

\* [ ] Les Models s’initialisent eux-mêmes, avec des évènements qui indiquent leur chargement effectif et récupère les données à mettre dedans (en théorie, l’interface est « statique »)