Complément monogame – Tuto gestion de scènes

La classe Game sera le chef d'orchestre : elle dispose de toutes les scènes et décide d'afficher celle en question en fonction des interactions ou de la logique du jeu.

Exemple simple : Apprendre à faire un changement de scène simple avec les flèches gauche et droite.

1. Créer un nouveau projet GestionScenes.

2. **Ajouter le package de Monogame.Extended afin d'utiliser les scenes.**

3. Créer un gestionnaire de scènes dans Game1.cs, ajoutez le champ suivant :
   ```
   private readonly ScreenManager _screenManager;
   ```

4. Attention : l'objet spriteBatch habituel qui sert à dessiner doit être publique afin de pouvoir l'utiliser et dessiner dans toutes les scènes.
   ```
   public SpriteBatch SpriteBatch {get; set;}
   ```
   Rem : ici, c'est une contraction pour faire un champ + propriété publique sans traitements. Ainsi on pourra utilizer cette propriété dans les classes de scenes pour dessiner ou écrire.

5. Dans le constructeur de Game1, initialiser ce gestionnaire :
   ```
   _screenManager = new ScreenManager();
   Components.Add(_screenManager);
   ```

6. Créer une nouvelle classe nommée MyScreen1.cs héritant de GameScreen :

```
public class MyScreen1 : GameScreen
{
    private Game1 _myGame;
// pour récupérer une référence à l'objet game pour avoir accès à tout ce qui est
// défini dans Game1
    public MyScreen1(Game1 game) : base(game) {
        _myGame = game;
    }
    public override void LoadContent()
```

```
    {

        base.LoadContent();
    }
    public override void Update(GameTime gameTime)
    {}
    public override void Draw(GameTime gameTime)
    {
        _myGame.GraphicsDevice.Clear(Color.Red); // on utilise la reference vers
          // Game1 pour chnager le graphisme
    }
}
```

7. Faites de même avec MyScreen2.cs (changer la couleur d'arrière-plan en bleu dans Draw)

8. Dans la classe Game1.cs, définissez 2 champs : _myScreen1 de type MyScreen1 et _myScreen2 de type MyScreen2.

9. Dans la classe Game1.cs au sein de la méthode LoadContent, initialisez ces 2 champs :

```
_ myScreen1 = new MyScreen1(this); // en leur donnant une référence au Game
_ myScreen2 = new MyScreen2(this);
```

10. Dans la classe Game1.cs au sein de la méthode Update, détectez quand changer de scène :

```
KeyboardState keyboardState = Keyboard.GetState();
if (keyboardState.IsKeyDown(Keys.Left))
{
 _screenManager.LoadScreen(_myScreen1, new FadeTransition(GraphicsDevice,
Color.Black));
}
else if (keyboardState.IsKeyDown(Keys.Right))
{
_screenManager.LoadScreen(_myScreen2, new FadeTransition(GraphicsDevice,
Color.Black));
}
```