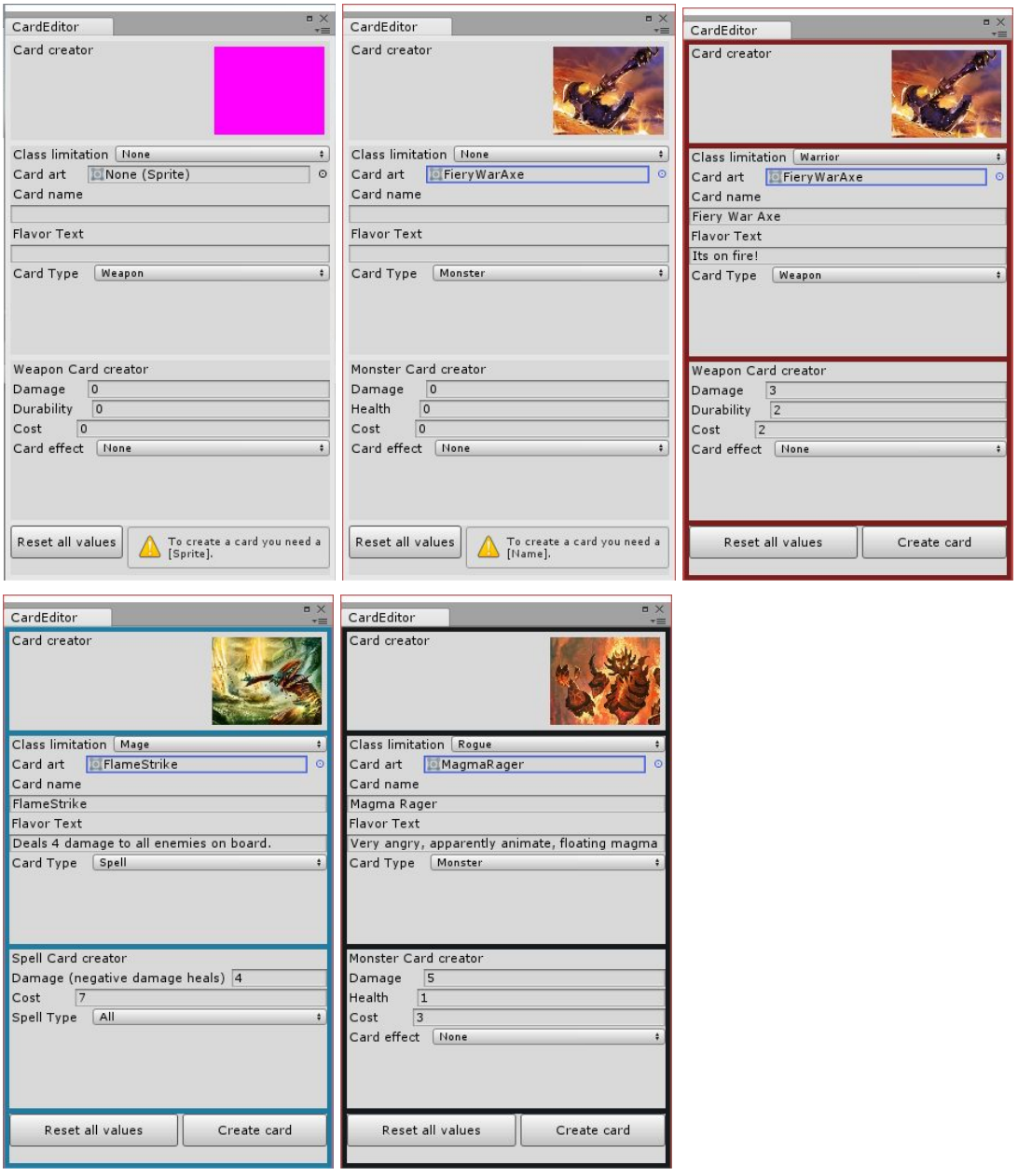
Kernmodule
Game-development blok 2

Vincent Creveld
3019866

Inhouds Opgave

Uitleg tool

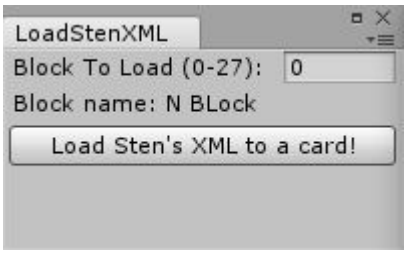
Main tool



Met deze tool kan je je eigen Hearthstone-achtige kaarten maken. Dit doe je door de invulvelden in te vullen.
Aan de hand van de geselecteerde class veranderd de trim van de tool.
Aan de hand van de geselecteerde kaart (weapon, monster, spell) wordt het onderste paneel aangepast. Dit is om de cutter van het scherm te minimaliseren.
Een kaart moet minimaal een sprite en naam hebben, anders kan je niet op de “create” knop klikken.

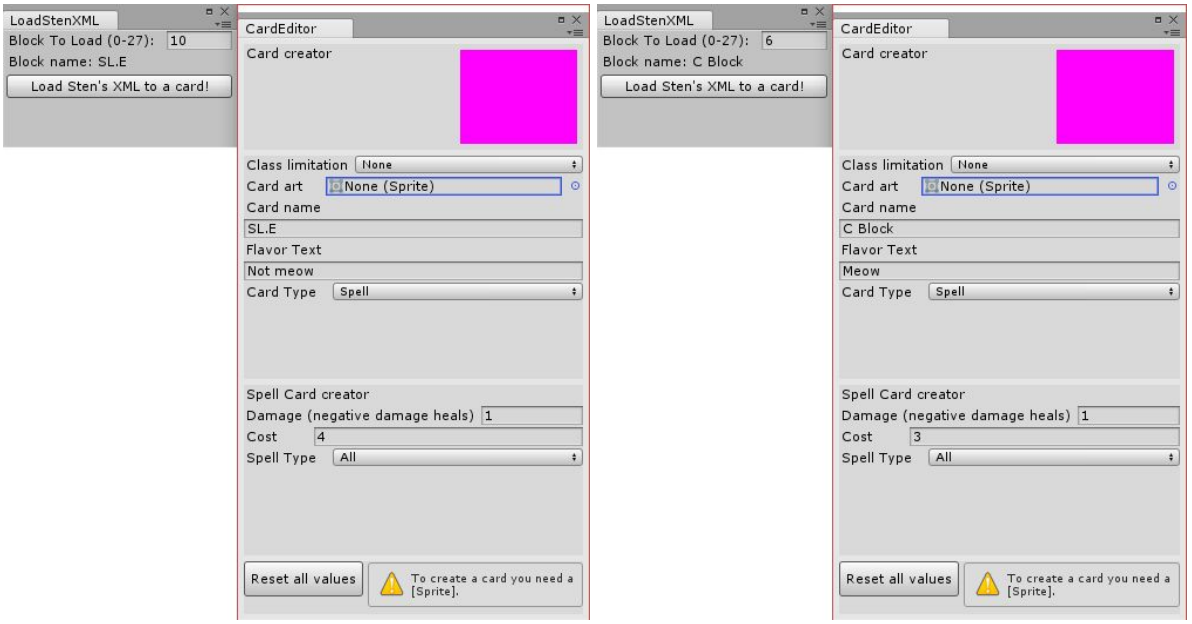
“Buddy”-systeem

De data is van Sten Duindam.



Dit is een aparte editor window om de XML data van Sten’s Block-creator tool om te zetten en er een kaart van te genereren.

Tool in werking:
https://gyazo.com/84f810b546f9229844148b54cbfe2346



Code snippets conversietool

```
public void LoadCardData() {
    XmlSerializer xmlSerializer = new XmlSerializer(typeof(BlockDataContainer));
    string path = resourcesPath + "/XML_Sten/Block_XML.xml";
    FileStream fileStream = new FileStream(path, FileMode.Open);
    container = xmlSerializer.Deserialize(fileStream) as BlockDataContainer;
    containerLength = container.createdBlockList.Count;
    fileStream.Close();
}

private void SetupReflection() {
    LoadCardData();
    CardDataReflection reflection = new CardDataReflection();
    BlockData loadedBlock = container.createdBlockList[blockToLoad];
    reflection.name = loadedBlock.name;
    reflection.FlavorText = loadedBlock.cat;
    reflection.CardType = loadedBlock.tier;
    reflection.health = loadedBlock.xSize;
    reflection.damage = loadedBlock.ySize;
    reflection.cost = loadedBlock.tier;
    SubmitReflection(reflection);
}

private void SubmitReflection(CardDataReflection reflection) ...
```

```
    switch(reflection.cardType) {
        case CardData.CardType.monster:
            CardEditor.selectedCard = CardEditor.SelectedCard.monster;
            CardEditor.monsterCard.dmg = reflection.damage;
            CardEditor.monsterCard.hp = reflection.health;
            CardEditor.monsterCard.name = reflection.name;
            CardEditor.monsterCard.sprite = reflection.sprite;
            CardEditor.monsterCard.cost = reflection.cost;
            CardEditor.monsterCard.flavorText = reflection.flavorText;
            break;
        case CardData.CardType.spell:
            CardEditor.selectedCard = CardEditor.SelectedCard.spell;
            CardEditor.spellCard.dmg = reflection.damage;
            CardEditor.spellCard.name = reflection.name;
            CardEditor.spellCard.sprite = reflection.sprite;
            CardEditor.spellCard.cost = reflection.cost;
            CardEditor.spellCard.flavorText = reflection.flavorText;
            break;
        case CardData.CardType.weapon:
            CardEditor.selectedCard = CardEditor.SelectedCard.weapon;
            CardEditor.weaponCard.dmg = reflection.damage;
            CardEditor.weaponCard.durability = reflection.health;
            CardEditor.weaponCard.name = reflection.name;
            CardEditor.weaponCard.sprite = reflection.sprite;
            CardEditor.weaponCard.cost = reflection.cost;
            CardEditor.weaponCard.flavorText = reflection.flavorText;
            break;
    }
}
```

```
//input cat value from XML
public int FlavorText {
    set {
        if(value == 0)
            flavorText = "Not meow";
        else
            flavorText = "Meow";
    }
}
```

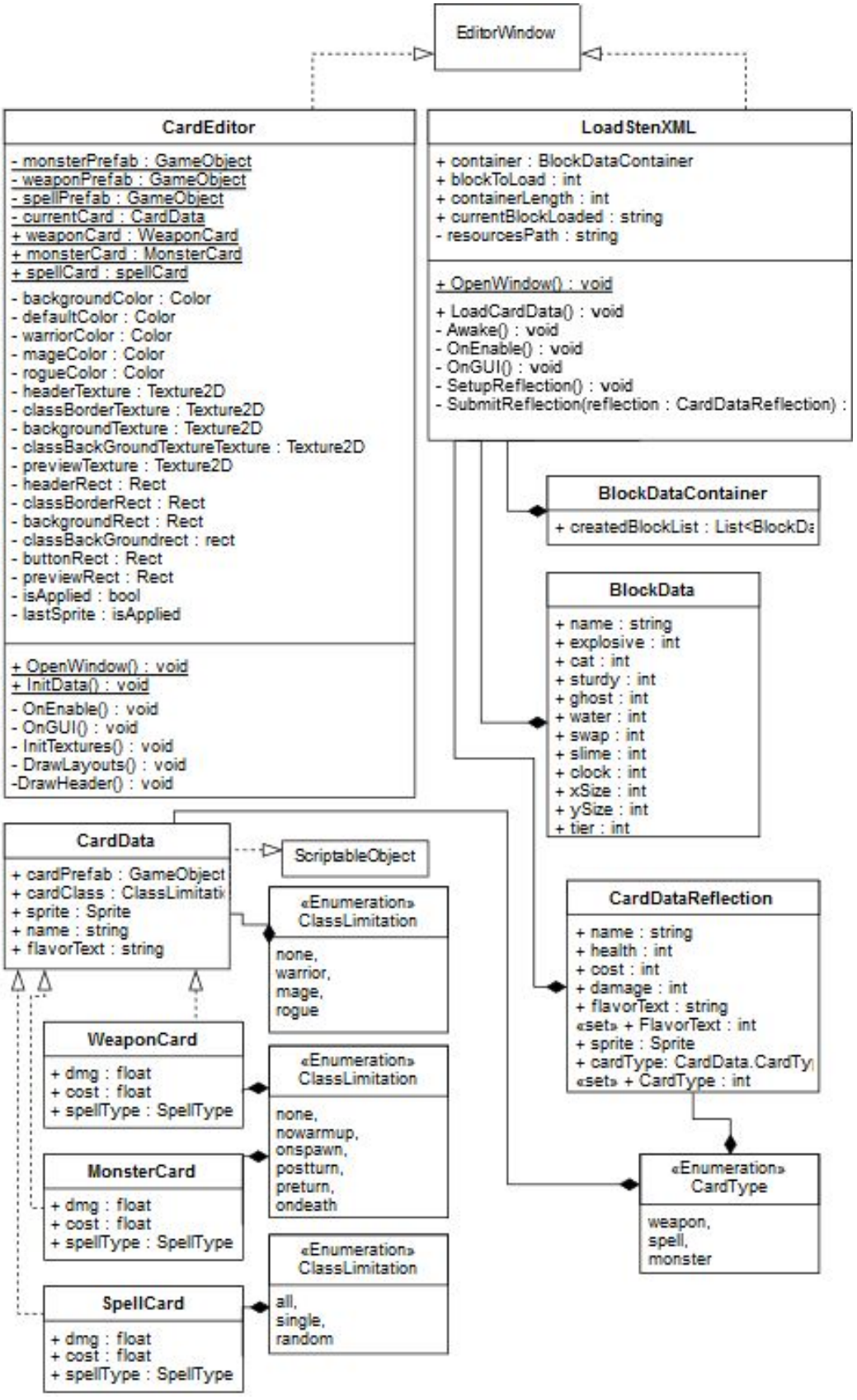
```
public CardData.CardType cardType;
public int CardType {
    set {
        switch(value) {
            case 1:
                cardType = CardData.CardType.monster;
                break;
            case 2:
                cardType = CardData.CardType.monster;
                break;
            case 3:
                cardType = CardData.CardType.spell;
                break;
            case 4:
                cardType = CardData.CardType.spell;
                break;
            case 5:
                cardType = CardData.CardType.weapon;
                break;
            case 6:
                cardType = CardData.CardType.weapon;
                break;
            default:
                cardType = CardData.CardType.monster;
                break;
        }
    }
}
```

Omdat Sten in zijn XML alleen een naam als string heeft staan en de rest als int, heb ik voor mijn flavorText zijn 0/1 int systeem als een boolean gebruikt. Om de cardType te bepalen heb ik aan de hand van de “tiers” van zijn blokken bepaald of een kaart een monster, weapon of spell is.

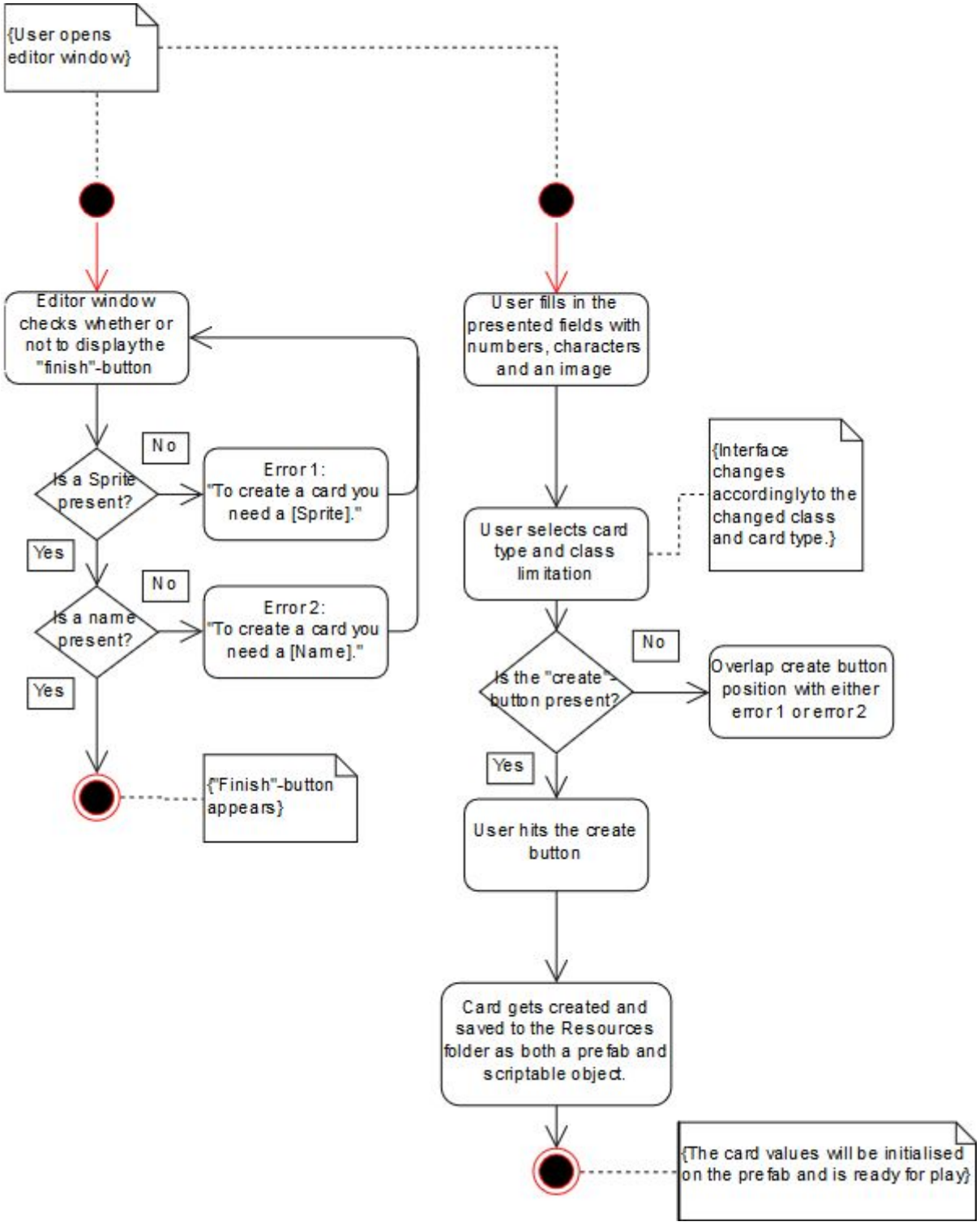
```
<BlockData Name="C Block">
  <explosive>0</explosive>
  <cat>1</cat>
  <sturdy>0</sturdy>
  <ghost>0</ghost>
  <water>0</water>
  <swap>0</swap>
  <slime>0</slime>
  <clock>0</clock>
  <xSize>1</xSize>
  <ySize>1</ySize>
  <tier>3</tier>
</BlockData>
```

Sten’s UML format

UML


EditorWindow
CardEditor
- monsterPrefab : GameObject
- weaponPrefab : GameObject
- spellPrefab : GameObject
- currentCard : CardData
+ weaponCard : WeaponCard
+ monsterCard : MonsterCard
+ spellCard : spellCard
- backgroundColor : Color
- defaultColor : Color
- warriorColor : Color
- mageColor : Color
- rogueColor : Color
- headerTexture : Texture2D
- classBorderTexture : Texture2D
- backgroundTexture : Texture2D
- classBackGroundTextureTexture : Texture2D
- previewTexture : Texture2D
- headerRect : Rect
- classBorderRect : Rect
- backgroundRect : Rect
- classBackGroundrect : rect
- buttonRect : Rect
- previewRect : Rect
- isApplied : bool
- lastSprite : isApplied
+ OpenWindow() : void
+ InitData() : void
- OnEnable() : void
- OnGUI() : void
- InitTextures() : void
- DrawLayouts() : void
-DrawHeader() : void
LoadStenXML
+ container : BlockDataContainer
+ blockToLoad : int
+ containerLength : int
+ currentBlockLoaded : string
- resourcesPath : string
+ OpenWindow() : void
+ LoadCardData() : void
- Awake() : void
- OnEnable() : void
- OnGUI() : void
- SetupReflection() : void
- SubmitReflection(reflection : CardDataReflection) :
BlockDataContainer
+ createdBlockList : List<BlockDa
BlockData
+ name : string
+ explosive : int
+ cat : int
+ sturdy : int
+ ghost : int
+ water : int
+ swap : int
+ slime : int
+ clock : int
+ xSize : int
+ ySize : int
+ tier : int
CardData
+ cardPrefab : GameObject
+ cardClass : ClassLimitati
+ sprite : Sprite
+ name : string
+ flavorText : string
ScriptableObject
«Enumeration»
ClassLimitation
none,
warrior,
mage,
rogue
CardDataReflection
+ name : string
+ health : int
+ cost : int
+ damage : int
+ flavorText : string
«set» + FlavorText : int
+ sprite : Sprite
+ cardType: CardData.CardTy
«set» + CardType : int
«Enumeration»
CardType
weapon,
spell,
monster
WeaponCard
+ dmg : float
+ cost : float
+ spellType : SpellType
«Enumeration»
ClassLimitation
none,
nowarmup,
onspawn,
postturn,
preturn,
ondeath
MonsterCard
+ dmg : float
+ cost : float
+ spellType : SpellType
«Enumeration»
ClassLimitation
all,
single,
random
SpellCard
+ dmg : float
+ cost : float
+ spellType : SpellType

Activity diagram



Links

Github:
https://github.com/VincentCreveld/KGDEVblok2

Youtube:
https://www.youtube.com/watch?v=WLIdWH5YV9k

“Buddy”-systeem tool in werking: Tool in werking:
https://gyazo.com/84f810b546f9229844148b54cbfe2346