Components

Un objet **Components** est un conteneur de données

Les données dans un objet **Component** ne peuvent pas être un autre objet **Component**

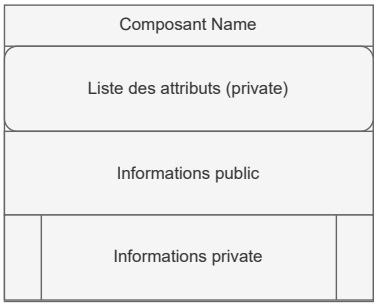
Toutes les classes représentant un objet **Component** doivent héritées de l'interface **IComponent**

Un composant représente des données *private* avec des fonctions pour les modifier appelées par des objets **Systems**

Color Legend

lack of design | designed | under development | developped

Box Legend



class parent communes à tous
IComponent

InputMapping	Position	MoveSpeed	Sprite
FIFO data - input	vector2d _position	float _moveSpeed	sf::Sprite
au constructeur - ajouter this au service *KeyboardInput* bool hasInput() void consumeNextInput()	Position(double x, double y) vector2d getPosition() void update(vector2d target)	MoveSpeed(float speed) float getSpeed() void applySpeedUp() void applySpeedDown()	void draw(Position currentPosition)

BackPack	Health	Lifetime	BombPower	Orientation
ArrayList<BombSlot > bombs int currentSelection	int _HP void* _onDeathTrigger	DateTime _start TimeSpan _lifeSpan void* _onEndOfLifeTrigger	int _damage int _range	enum {N, W, S, E}
void dropBomb(Position position) void moveSelection(int step)	Healt(int initialHealtPoints) void update(int damage) bool isDead() void triggerDeath()	LifeTime(DateTime start, TimeSpan duration) void update(TimeSpan lastUpdate) boolean isLifeOver() void triggerEndOfLife()	void applyPowerUp() void applyPowerDown()	

struct BombSlot {
IBomb bomb;
int count
}				

AnimatedSprite	HitBox
sf::Sprite spriteCount current elapsedTime	Collection<Square> _shape
void updateElapsedTime(TimeSpan) void draw(Position currentPosition)	bool hasCollision(Position currentPosition, HitBox target, Position targetPosition)