Фича-компонентный подход

Как сделать игровые фичи более гибкими

Сергей Соложенцев

Viber/Telegram: +375293408979

Масштаб проблемы

Почему не наследование?

Пример игры

Игра 3 в ряд. Нужно собрать в цепочку не менее 3 камней.

История решения

All-in-one

```
int m_row;
int m_col;
MatchElementType m_type;
MatchElementKind m_kind;
int m_hits;
CMatchElement* m_pOwner;
CMatchElement* m_pCreatedElement;

bool m_enabled;
bool m_hasFarm;
bool m_enableToChange;
bool m_hasFire;
int m_ignoreCurseTurns;
bool m_underCurseSpread;

MatchElementType m_bombType;
int m_bellIndex;
bool m_selectable;
int m_updateHits;
CPooledVector<Point> m_dependentElements;
MatchElementType m_secondType;
int m_previousMoveDirection;
bool m_multipleHits;
PiranyaValues m_piranyaValues;

bool m_willDelete;

int m_scoreMultiplier;
bool m_hasKey;
bool m_hasWatch = false;
```

Мастер на все руки плюсы и минусы

Все в одном месте, не надо искать

“Магический” тип элемента

Очень много ненужных данных для каждого объекта

переиспользование одних переменных для разных фич, что может влиять на сочетание фич

сложно отделить нужное от устаревшего

Попытка номер 2

Компонентный подход

```
private:
        CPointerArray<CElementComponent> m_components;

public:
        void addComponent(CElementComponent* pComponent);
        CElementComponent* getComponent(ElementComponentType
type);
        bool hasComponent(ElementComponentType type);
```

Что изменилось

Данные для каждой фичи отделены

Можно легко убирать и добавлять фичи в конкретные элементы

Игровая логика перегружена поддержкой всех фич. В ней сложно разбираться и поддерживать.

Текущее решение

Основные понятия

Ключевой объект

Точка расширения

Компонент

Текущее решение

Фича-компонентный подход

Ключевые принципы

- Каждый ключевой объект является контейнером компонентов
- Ключевой объект может иметь точки расширения функциональности
- Ключевые объекты могут иметь несколько контейнеров компонентов разных типов
- Компоненты могут использовать другие компоненты получая их из ключевого объекта


Начало
Начальная информация
Компонент 4
Компонент 1
TP
Компонент 3
Компонент 2
Преобразованная информация
следующая точка расширения
или финальная обработка

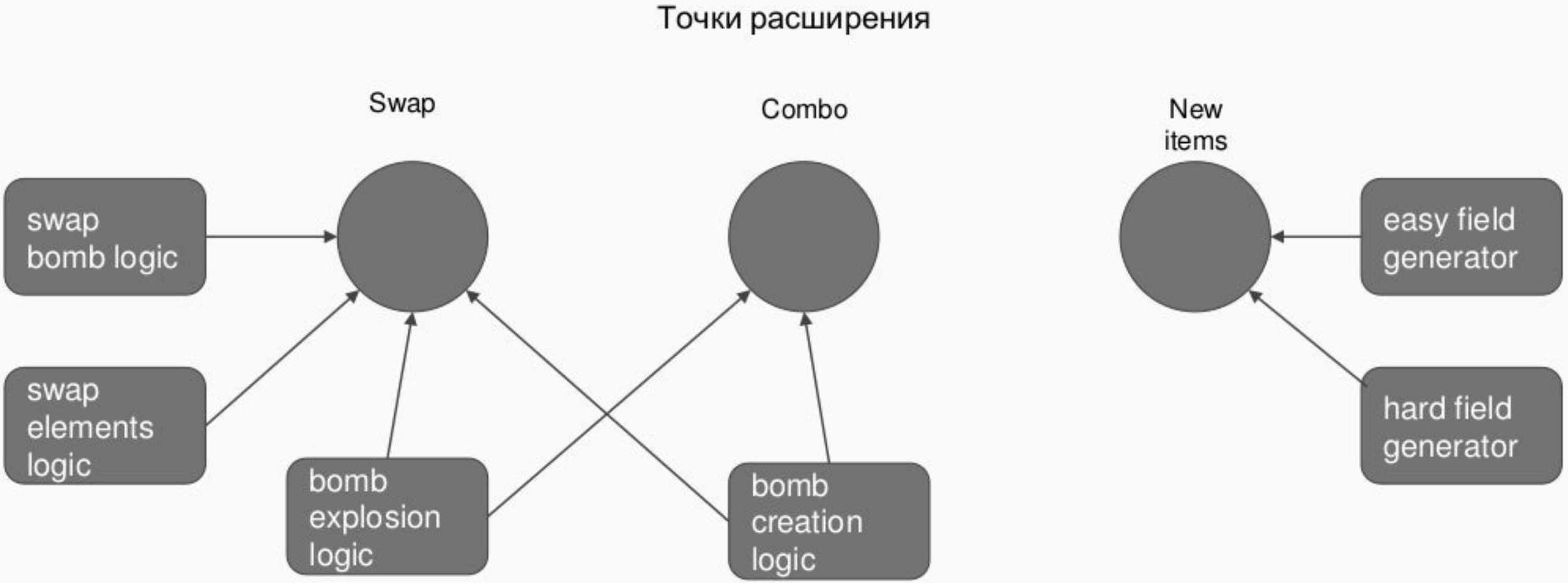
Примеры преобразования

- Добавление / удаление / изменение элементов
- Изменение состояния ключевых объектов
- Информирование внешних систем (сбор статистики например)

Проблемы

Проблема	Вариант решения
Требуется определенный порядок обработки компонентов	Компоненты добавляются в контейнер в определенном порядке (например по численному значению типа)
Нужно прекращать обработку по сигналу компонента	Поддерживать возвращаемое значение в точке расширения, чтобы можно было прекратить обработку
Требуется отменить обработку по сигналу компонента	Сделать еще одну точку расширения, в которой можно отменить обработку, ничего не меняя

Пример


Точки расширения
Swap
Combo
New items
swap bomb logic
swap elements logic
bomb explosion logic
bomb creation logic
easy field generator
hard field generator

Реализация точек расширения

Delegate

Интерфейс

Туториал

Игра с туториалом

Стандартный компонент 1

Стандартный компонент 2

Генерация специального поля

Подстветка цепочки

Игра без туториала

Стандартный компонент 1

Стандартный компонент 2

Будущее

Какие есть идеи для улучшения

Универсальный движок для игры

Игра как набор компонентов, которые взаимодействую между собой посредством понятных интерфейсов.

Позволяет включать и разделять многие компоненты между разными играми, например набор очков или PvP

Спасибо!

Сергей Соложенцев

Viber/Telegram: +375293408979