**COOP COMMANDERS**

Исправленная версия стандартной библиотеки совместного режима

Оглавление

[Используемые термины 2](#_Toc201240702)

[Стандарты при создании данных 3](#_Toc201240703)

[Разделение зависимостей командиров 3](#_Toc201240704)

[Использование наборов данных (Data Collection) 3](#_Toc201240705)

[Создание единиц данных 3](#_Toc201240706)

[Работа с текстом 4](#_Toc201240707)

[Остальная структура данных 4](#_Toc201240708)

[Основные положения мастеров (Wizards) 5](#_Toc201240709)

[Общее 5](#_Toc201240710)

# Используемые термины

**Единица данных** – один из основных объектов игровых данных таких как “Единица”, “Способность”, “Улучшение”, “Набор данных”

**Дополнительная единица данных** – Вложенные объекты в основную единицу данных. Например: “Оружие”, “Эффект”, “Алгоритм”, “Агент”, “Модель”, “Звук” и прочие.

**Родитель** – единица данных, от которой должны создаваться другие единицы данных.

# Стандарты при создании данных

Здесь рассмотрены все необходимые рекомендации, для обеспечения стандартизации данных проекта. Следование этим рекомендациям, обеспечивает однородность наборов библиотек командиров

## Разделение зависимостей командиров

1. Все данные командиров НЕ должны явно связываться друг с другом, и ДОЛЖНЫ быть в своих собственных зависимых модификациях *(dependent mod).*
2. Каждая модификация должна исправно работать без любых других зависимых модификаций.
3. Каждая зависимая модификация командира должна называться по стандарту “Commander-{NameCommander}.SC2Mod”.
4. Каждая зависимая модификация в репозитории проекта “COOP-COMMANDERS”, должна быть разобрана на компоненты мода.
5. Каждая зависимая модификация командира, должна иметь только одну подключенную зависимость, такую как “COOPCore.SC2Mod”

## Использование наборов данных (Data Collection)

1. При добавлении новой единицы данных, необходимо создать для неё набор данных (Data Collection).
2. ID новой единицы данных, должно совпадать с ID набора данных (Data collection).

## Создание единиц данных

1. **Использование родителей.** При создании единицы данных, следует использовать соответствующего родителя. Например: для создания новой **наземной единицы терранов** следует использовать родителя **Commander Unit - Ground (Terran)**. Это касается всех единиц данных, будь то единица, Актор, Модель, Звук, Набор данных и прочие
2. **Изолирование.** Каждая **единица данных** должна быть спроектирована изолированно относительно других **единиц данных**. Это подразумевает, что каждая **единица данных** не должна иметь жестких зависимостей от других **единиц данных**.
3. **Использование наборов данных.** Все **единицы данных** необходимо организовать в отдельные **наборы данных**. Набор данных включает в себя все связанные данные данной единицы данных. Это упрощает управление данными, позволяет легко копировать и изменять объекты, а также единообразно настраивать данные через паттерны наборов данных.
4. **Использование категорий редактора.** Чтобы упростить поиск, редактирование и поддержку данных, связанных с каждым командиром, необходимо использовать специальные категории редактора (COMMANDER - Raynor, COMMANDER - Kerrigan, COMMANDER - Artanis и т.д.). При создании новых данных, всегда выбирайте соответствующую категорию командира. Это значительно облегчит поиск данных, относящихся к конкретному командиру, а также позволит легко определить, какие данные требуют обновления при внесении изменений в геймплей соответствующего командира.

## Работа с текстом

1. Любое упоминание единицы данных в игровых подсказках, выводить при помощи

**<d stringref=”Catalog,Entry,Field”/>**

## Остальная структура данных

1. Каждый командир является **отдельной расой.** Все единицы командира принадлежат его расе
2. ID расы командира должно начинаться с **“COOP”**, далее идёт **имя** командира. Пример: для командира **Рейнора**, создана раса *“COOPRaynor”*.

# Основные положения мастеров (Wizards)

Эта глава посвящена разработке мастеров (визардов) для набора библиотек

## Общее

1. Каждый мастер должен строго соблюдать все перечисленные стандарты
2. Управление мастером должно быть простым для начинающих разработчиков
3. В каждом мастере должна быть прописана четкая инструкция внутри самого мастера, с помощью тега **<instructions>< /instructions >.** При необходимости прописывать инструкцию для других страниц мастера.
4. Максимально сокращать количество страниц мастера. Но и не перегружать первую страницу