**4. GAME OVER WIDJET. PLAYER STATS**

1. Будем создавать виджет для GameOver. Что добавили в гейм худ и как изменили BeginPlay? Как создали BP-виджет для GameOver?

2. Создаем дополнительный виджет для статистики – как настроили?

3. Куда теперь добавили виджет для статистики? Что за элемент еще добавили, который будет динамически заполняться статистикой?

4. Переходим к коду. Создадим два виджета – один будет базовым классом для строчки статистики, второй – для GameOver. Что определяем в STUPlayerStatRowWidget (5 проперти и 5 функций)?

5. Добавили два проперти и 3 функции в GameOverWidget – какие? Какое в них определили поведение? Функция для добавления дочернего элемента в UI-компонент? Функция для удаления дочерних элементов из UI-компонента? Какая есть функция в базовом PlayerState для имени персонажа и что содержит по умолчанию для разных контроллеров? вспомогательную функцию в STUUtils – зачем?

6. В функции CreateTeamsInfo внесем изменения (в гейм моде) – какие? Как теперь настраиваем ранее созданные BP-виджеты?

7. Какой визуальный недочет у нас сейчас есть в определенных условиях у нашей таблицы (критический недостаток, не просто «некрасивенько»? Как пофиксили?

1. Займемся созданием виджета, который появляется при окончании игры, на котором будет отображать статистику по всем игрокам.

Добавляем виджет в гейм худ:

Изображение выглядит как текст, внутренний, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Добавляем BP-виджет GameOver – копируем ассет WBP\_GamePause и называем WBP\_GameOver. Удалим кнопку, поменяем надпись на GAME OVER. Реперентим на UserWidjet. В GameHUD указываем наш новый виджет для GameOver.

2. Теперь выведем статистику. Для этого нам понадобится дополнительный виджет – строчка со статистикой по одному игроку, а в классе GameOver-виджета мы будем динамически создавать нужное нам количество строчек. Для этого создали новый BP-виджет WBP\_PlayerStatRow.

У него следующая структура и вид:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст, электроника, цепь

Автоматически созданное описание

Мы добавили дополнительный элемент Image, который будет подсвечивать нашего персонажа. Он должен быть растянут на всю ширину HorizontalBox и находится слоем ниже. UI элемент, который поможет это сделать – Overlay (через ПКМ->WrapWith по HorizontalBox). Overlay позволяет группировать различные UI элементы слоями. Мы перетащили после этого Image в Overlay. Картинке ставим Visibility в Hidden.

3. Строчка для статистики одного игрока готова – возвращаемся в GameOverWidjet и добавляем сюда наш виджет для строки статистики. Данный виджет будет использоваться так же в качестве заголовка. Мы его сейчас добавили – динамически с ним ничего не будет происходить. Добавляем спейсер еще один для разделения заголовка и результатов и вертикальный бокс, в котором будет храниться вся таблица. В данном боксе данные мы будем создавать динамически.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст, флэш-память, устройство, датчик

Автоматически созданное описание

4. Переходим к коду. Создадим два виджета – один будет базовым классом для строчки статистики, второй – для GameOver. Определяем в STUPlayerStatRowWidjet:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

5. Переходим в ЗФ GameOverWidjet:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, экран

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Функция добавления дочернего элемента в UI-компонент – AddChild.

Предусмотрели ситуацию и удаляем с помощью ClearChildren ранее созданные строчки, если они были.

В PlayerState в базовом классе существует функция GetPlayerName, то есть в PlayerState существует переменная, которая отвечает за имя игрока. Для AIController это обычно пустая строчка, а для PlayerController – имя компьютера.

Создали при этом вспомогательную функцию в STUUtils для перевода из инта в FText:

Изображение выглядит как текст

Автоматически созданное описание

6. В функции CreateTeamsInfo установим имя для PlayerState.

Изображение выглядит как текст

Автоматически созданное описание

Теперь надо поменять родительский класс у GameOver – установить STUGameOverWidjet, настроить имя для VerticalBox и перейти в Graph, где настроить класс для нашего PlayerStatRowWidjetClass.

Для WBP\_PlayerStatRow делаем аналогичные действия.

7. Все хорошо, только если игроков у нас не слишком много. В противном случае у нас таблица вылезет за пределы экрана. Чтобы это исправить, обрамим PlayerStatBox в специальный UI-элемент ScrollBox – он маскирует контент и добавляет полосу прокрутки. Так как у нас сейчас стоит и всего VerticalBox (родительского) параметр SizeToContent, то ничего не изменится. Для исправления этого обрамляем наш уже Scroll Box в Size Box и задать ему явно высоту – параметр **MaxDesiredHeight** (то есть не просто высоту, а максимальную высоту – как max-width в CSS). Появился ползунок:



Сделаем так, чтобы полоска скролл-бара была видна постоянно, а так же уберем тени. И еще добавили спейсер, чтобы он отделял последнее текстовое поле от скролл-бара.