**TextColor – не трогать**

**TextColor – ожидает создания**

**TextColor – под вопросом**

**BP\_BaseActor –** базовый Actor, который можно создать на сцене**.**

**Методы:**

**Category [Debug]:**

**public:**

* **DebugLog** – дебаг сообщений на экран

**Category [Default]:**

**public:**

* **HideMe** – прячет игровой объект из виду (лучше переопределять, потому что для разных объектов происходит разный процесс спрятывания. Т.е. одни объекты могут иметь меш, другие могут иметь коллайдер, третьи могут иметь и то и другое и несколько штук. Например 10 мешей, 5 коллайдеров и еще что-то).
* **UnHideMe** – показывает спрятанный игровой объект. Обратное действие к HideMe. Тоже лучше переопределить и написать ту же логику что в HideMe, только наоборот.
* **ReloadMe** – перезапускает объект (восстанавливает в исходное положение). Характерно для перезапуска таски. Нужно переопределять и писать логику исходного положения. Чтоб при вызове этой функции объект был таким как будто только что заспавлен.
* **DestroyMe** – удаляет игровой объект из массива ITickable, если нужно. А так-же удаляет его со сцены.

**Переменные:**

**Category [Default]:**

**public:**

* **ID\_Name – имя объекта, который спавнится на сцене. Нужно для того чтоб при взаимодействии с этим объектом, мы могли внести изменения в файл сохранения, в котором будет словарь заспавленных объектов и доступ к определенному объекту можно получить по его имени**.
* **ID\_Actor** – индекс объекта, который спавнится на сцене (так как все объекты при спавне добавляются в массив, то удобно управлять ими из массива через индекс, например управлять видимостью при восстановлении из сохранения (объект с таким-то индексом видимый или нет на сцене)). Так как заспавленные объекты хранятся в словаре **SpawnedActorsMap**, к которому можем получить доступ по строке, то чтоб получить по индексу, нам нужно получить **Values** этого словаря и потом получать доступ по индексу.
* **MainGameMode** – ссылка на главный **GameMode**
* **CanTick** – если true, то при создании объекта, он добавляется в массив **ITickable**, что в компоненте **TickManagerComponent** (см. доку этого компонента) и для него будет выполняться Tick.
* **IsActorActive** – активен ли объект на сцене или нет (нужен для сохранения, если активен, то просто создаем этот объект на сцене и все, а если не активен, то создаем и прячем). По умолчанию **true**.
* **StartActorTransform** – начальный трансформ этого объекта после спавна (то место где он заспавнился). Нужно для перезагрузки таски например.

**Category [Components]:**

**public:**

* **SceneRoot** – пустышка, центр объекта.