|  |
| --- |
| Feldfeste Studios |
| **Castle Age API** |
| Design Document |

|  |
| --- |
| Dr Zachary Parker |
| Version Number: 0.2 |

# Notes

## Scope of Use

Content and code may be used in castle age the game, and in tutorials by Zachary, as long as those tutorials do not reveal privilege game information (e.g., how certain assets are made, the full scope of the AI, unique game mechanics). This limit does not include game mechanics common to the genre.

## Naming Conventions & Issues In Documentation

Sometimes the use of language and terms are no consistent in the notes section. For example, this document will often refer to “GameState” or “the game state” and in neither case is this the generic GameState/GameStateBase; this is a reference to RTSGameState. Likewise, some variables (e.g., DayOfTheWeek) are sometimes referenced spelled the same and sometimes they are broken into individual words (e.g., Day Of The Week). **Again this only occurs in the notes column, all listings are correct in the name and type columns.** I will be working to update this as I go.

## API in progress

Please note that this API is being written at the same time as development work continues. This means that some classes may be missing or have incorrect information (e.g., list functions that are going to be deprecated and used as place holders). Any version of this document prior to version 1.0 (see cover page) should be considered an API in progress. **If you are using this after version 1.0 make sure that the version of code you are using matches the API you are using.**

## Organization of API

This API is organized by the folder structures of the project. The folder structure is alphabetical. The classes are then listened alphabetically. The top left of the page will tell you which section you are in (like here it says ‘*Notes*’).

**Table of Contents**

[Notes 2](#_Toc89506322)

[Scope of Use 2](#_Toc89506323)

[Naming Conventions & Issues In Documentation 2](#_Toc89506324)

[API in progress 2](#_Toc89506325)

[Organization of API 2](#_Toc89506326)

[Version History 6](#_Toc89506327)

[BuildingMasterBase 7](#_Toc89506328)

[References 7](#_Toc89506329)

[Remarks 7](#_Toc89506330)

[Variables 7](#_Toc89506331)

[Constructors 7](#_Toc89506332)

[Functions 7](#_Toc89506333)

[SunRotationBase 8](#_Toc89506334)

[References 8](#_Toc89506335)

[Remarks 8](#_Toc89506336)

[Variables 8](#_Toc89506337)

[Constructors 8](#_Toc89506338)

[Functions 8](#_Toc89506339)

[CustomMovementComponent 9](#_Toc89506340)

[References 9](#_Toc89506341)

[Remarks 9](#_Toc89506342)

[Variables 9](#_Toc89506343)

[Constructors 9](#_Toc89506344)

[Functions 9](#_Toc89506345)

[Enums 12](#_Toc89506346)

[Remarks 12](#_Toc89506347)

[BuildingEnum 13](#_Toc89506348)

[References 13](#_Toc89506349)

[EBuildingType 13](#_Toc89506350)

[EBuildingName 13](#_Toc89506351)

[ResourcesEnum 14](#_Toc89506352)

[References 14](#_Toc89506353)

[EProducedResources 14](#_Toc89506354)

[EFoodResources 14](#_Toc89506355)

[EVisualResources 14](#_Toc89506356)

[CameraPawnInterface 15](#_Toc89506357)

[References 15](#_Toc89506358)

[Remarks 15](#_Toc89506359)

[Functions 15](#_Toc89506360)

[GameStateInterface 16](#_Toc89506361)

[References 16](#_Toc89506362)

[Remarks 16](#_Toc89506363)

[Functions 16](#_Toc89506364)

[GameTimeInterface 17](#_Toc89506365)

[References 17](#_Toc89506366)

[Remarks 17](#_Toc89506367)

[Functions 17](#_Toc89506368)

[CamweraPawn 18](#_Toc89506369)

[References 18](#_Toc89506370)

[Remarks 18](#_Toc89506371)

[Variables 18](#_Toc89506372)

[Constructors 18](#_Toc89506373)

[Functions 19](#_Toc89506374)

[Interface Related Functions 19](#_Toc89506375)

[RTSPlayerController 20](#_Toc89506376)

[References 20](#_Toc89506377)

[Remarks 20](#_Toc89506378)

[Variables 20](#_Toc89506379)

[Constructors 20](#_Toc89506380)

[Functions 20](#_Toc89506381)

[Interface Related Functions 22](#_Toc89506382)

[RTSGameInstance 24](#_Toc89506383)

[References 24](#_Toc89506384)

[Remarks 24](#_Toc89506385)

[Variables 24](#_Toc89506386)

[Constructors 24](#_Toc89506387)

[Functions 24](#_Toc89506388)

[RTSGameState 25](#_Toc89506389)

[References 25](#_Toc89506390)

[Remarks 25](#_Toc89506391)

[Game Speed Breakdown 25](#_Toc89506392)

[Variables 25](#_Toc89506393)

[Constructors 26](#_Toc89506394)

[Functions 27](#_Toc89506395)

[Interface Related Functions 28](#_Toc89506396)

[UnitMasterBase 29](#_Toc89506397)

[References 29](#_Toc89506398)

[Remarks 29](#_Toc89506399)

[Variables 29](#_Toc89506400)

[Constructors 29](#_Toc89506401)

[Functions 29](#_Toc89506402)

[GameClockWidget 30](#_Toc89506403)

[References 30](#_Toc89506404)

[Remarks 30](#_Toc89506405)

[Variables 30](#_Toc89506406)

[Constructors 30](#_Toc89506407)

[Functions 30](#_Toc89506408)

[Interface Related Functions 30](#_Toc89506409)

[MainUIBase 31](#_Toc89506410)

[References 31](#_Toc89506411)

[Remarks 31](#_Toc89506412)

[Variables 31](#_Toc89506413)

[Constructors 31](#_Toc89506414)

[Functions 31](#_Toc89506415)

[Interface Related Functions 31](#_Toc89506416)

# Version History

|  |  |
| --- | --- |
| **Version Number** | **Last Updated** |
| 0.1 | 4 December 2021 |
| 0.2 | 4 December 2021 |

# BuildingMasterBase

## References

|  |  |
| --- | --- |
| Header |  |
| Source |  |

## Remarks

Lorem Ipsum

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ABuildingMasterBase () | Sets the default values |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

# SunRotationBase

## References

|  |  |
| --- | --- |
| Header |  |
| Source |  |

## Remarks

Lorem Ipsum

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ABuildingMasterBase () | Sets the default values |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

# CustomMovementComponent

## References

|  |  |
| --- | --- |
| Header | Core/Components/CameraMovementComponent.h |
| Source | Private/Core/Components/CameraMovementComponent.cpp |

## Remarks

This class is the camera pawn’s custom movement component. It implements the movement controls used by the player. While the methods for movement and zoom are here, they called in the CameraPawn.

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| Private | Bool | bDisableCameraMovement | Disables camera movement |
| Private | Bool | bDisableEdgeScoll | Disables edge scrolling – used when the player is rotating the camera |
| Private | Float | CornerEdgeScrollSpeed | Do not modify value. This is calculated for when edge scrolling is happening in two areas (e.g., top and left) so that the speed does not double |
| Private | Float | DefaultMovementSpeed | The base speed at which the camera pawn will move |
| Private | Float | DeltaTime | Delta time (do not modify) |
| Private | Float | DeltaZoomAmount | How much the camera will move in or out on the curve |
| Private | Float | MovementSpeedModifier | A multiplier used to speed the camera movements up - Defaults to 1, when used sets to 2. |
| Public | APlayerController\* | PlayerControllerRef | A reference to the player controller |
| Private | Float | ZoomAmount | How far zoomed in or out the camera is. |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | UCameraMovementComponent() | Sets the default values |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Public | Void | BasicMovementControl(float AxisValueX, float AxisValueY) | Moves the camera on the X and Y planes. |
| Protected | Void | BeginPlay() | The override of AActor begin play, gets and sets the PlayerControllerRef |
| Public | Void | CameraReset() | Returns the camera to the default rotation (pan) and zoom |
| Public | Void | EdgeScroll() | Moves the camera if the mouse is on the edges of the screen |
| Public | FVector | GetCameraLocation() | Returns the location of the camera pawn |
| Public | FRotator | GetCameraRotation() | Returns the camera pawn’s rotation |
| Public | Float | GetCurrentMovementSpeed() | Returns current value for DefaultMovementSpeed |
| Public | FVector | GetFaceDirection() | Returns which direction the camera pawn is facing |
| Public | Float | GetSpeedModifier() | Returns current value for MovementSpeedModifier |
| Public | Bool | IsCameraDisabled() | Returns if the camera is currently able to move or not (bDisableCameraMovement) |
| Public | Bool | IsEdgeScrollDisabled() | Returns if the camera can edge scroll (bDisableEdgeScoll) |
| Public | Void | PanCamera(float RotationAmount) | Rotates the camera |
| Public | Void | ResetPan() | Returns the camera to default rotation |
| Public | Void | ResetZoom() | Returns the camera to the default zoom amount |
| Public | Bool | SetCameraDiabled(bool bDisableCamera) | (Dis)enable camera movement |
| Public | Bool | SetDisableEdgeScroll (bool bDisableScroll) | (Dis)enable edger scroll |
| Public | Float | SetMovementSpeed(float MovementSpeedAdjustment) | Set DefaultMovementSpeed to the MovementSpeedAdjustment amount |
| Public | Float | SetMovementSpeedModifier(float ModifierAmount) | Set Speed Modifier (MovementSpeedModifier) via player controller (passes in value of 1 or 2) |
| Public | void | Tick(float DeltaTime, ELevelTick TickType, FActorComponentTickFunction\* ThisTickFunction) | Override of the tick function. Sets DeltaTime, checks zoom, and calls zoom related events |
| Public | Void | ZoomIn() | Zooms the camera in (shorten spring arm, rotate camera, and decrease FoV) |
| Public | Void | ZoomOut() | Zooms the camera out (lengthens spring arm, rotate camera, and increase FoV) |

# Enums

## Remarks

All Enums stored in the Enums folder are meant to be globally used enums.

# BuildingEnum

ENUMS related to buildings

## References

|  |  |
| --- | --- |
| #include | "Core/Enums/BuildingEnums.h" |

## EBuildingType

|  |  |
| --- | --- |
| **C++ value** | **Display name** |
| Communal | Communal |
| Infrastructure | Infrastructure |
| Agricultural | Agricultural |
| Artisanry | Artisanry |
| Wooden | Wooden |
| Stone | Stone |
| Warfare | Warfare |

## EBuildingName

|  |  |
| --- | --- |
| **C++ value** | **Display name** |
| - | - |

# ResourcesEnum

ENUMS for all resources

## References

|  |  |
| --- | --- |
| #include | "Core/Enums/ResourcesEnum.h" |

## EProducedResources

|  |  |
| --- | --- |
| **C++ value** | **Display name** |
| - | - |

## EFoodResources

|  |  |
| --- | --- |
| **C++ value** | **Display name** |
| - | - |

## EVisualResources

|  |  |
| --- | --- |
| **C++ value** | **Display name** |
| - | - |

# CameraPawnInterface

## References

|  |  |
| --- | --- |
| #include |  |

## Remarks

Lorem Ipsum

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

# GameStateInterface

## References

|  |  |
| --- | --- |
| #include |  |

## Remarks

Lorem Ipsum

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

# GameTimeInterface

## References

|  |  |
| --- | --- |
| #include |  |

## Remarks

Lorem Ipsum

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

# CamweraPawn

## References

|  |  |
| --- | --- |
| Header | Core/Player/CameraPawn.h |
| Source | Private/Core/Player/CameraPawn.cpp |

## Remarks

This class is the actor that houses the camera itself, and does the collision checks for the camera. **This case is fully implemented in blueprint (**BP\_)**!**

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| Public | UCameraMovementComponent\* | PawnMovementComponent | References to the CameraMovementComponent associated with this camera pawn. |
| Protected | USphereComponent\* | CollisionSphere | Root component of this actor, used to test for collision with boundaries of the map |
| Protected | UCameraComponent\* | PlayerCamera | The actual camera, attached to CameraBoom |
| Protected | USpringArmComponent\* | CameraBoom | A spring arm |
| Protected | Float | DefaultZoomDistance | Default length of CameraBoom, used to determine the default zoom distance |
| Protected | FRotator | DefaultCamerRotation | Default rotation of CameraBoom |
| Protected | Float | DefaultFieldOfView | Default FoV (should be 90) |
| Protected | UCurveFloat\* | CurveZoomArmLength | The curve graph used to adjust CameraBoom when zooming in and out (loaded in from a UASSET) |
| Protected | UCurveFloat\* | CurveZoomRotation | The curve graph used to adjust how much this actor rotates the camera towards or away from the ground when zooming in and out (loaded in from a UASSET) |
| Protected | UCurveFloat\* | CurveZoomFOV | The curve graph used for adjusting how much the FoV lowers when zooming in or increase when zooming out (loaded in from a UASSET) |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ACameraPawn() | Sets the default values, including root component and camera set up. Also loads in the curves being used |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Public | UCameraComponent\* | GetCamera() | Returns the camera component |
| Protected | Void | BeginPlay() | Nothing special just a blank override |
| Public | Void | Tick(float DeltaTime) | Runs AdjustCameraHeight |
| Public | USpringArmComponent\* | GetCamerBoom() | Returns the spring arm component |
| Public | Float | GetCurrentArmLength() | Returns the length of the spring arm |
| Public | FRotator | GetCurrentRotation() | Returns current rotation of spring arm |
| Public | Float | GetCurrentFieldOfView() | Returns current FoV value |
| Public | Void | SetArmLnength(float InZoomAmount) | For zoom events, sets the spring arm length |
| Public | Void | SetArmRotation(float InZoomAmount); | For zoom events, sets the rotation of the spring arm |
| Public | Void | SetFieldOfView(float InZoomAmount) | For zoom events, sets the camera’s FoV |
| Public | Void | AdjustCameraHeight() | Moves the entire actor up or down based on the landscape – this is how the actor responds to changes in the terrain. |

## Interface Related Functions

Remember these functions are defined with ‘\_Implementation’ and called with ‘Execute\_’

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Public | Void | CallSetArmlength(float Value) | Interface event that calls SetArmLength (used for communication between pawn, custom movement component, and player controller) |
| Public | Void | CallSetArmRotation(float Value) | Interface event that calls SetArmRotation (used for communication between pawn, custom movement component, and player controller) |
| Public | Void | CallSetFieldOfView(float Value) | Interface event that calls SetFieldOfView (used for communication between pawn, custom movement component, and player controller) |

# RTSPlayerController

## References

|  |  |
| --- | --- |
| Header | Core/Player/RTSPlayerControllerBase.h |
| Source | Private/Core/Player/RTSPlayerControllerBase.cpp |

## Remarks

This class is the player controller. It controls the movements of the pawn. It stores player data and has components associated with buildings, units, and resources.

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| Protected | ACameraPawn\* | CameraPawnRef | A pointer to the camera pawn |
| Protected | UCameraMovementComponent\* | CameraMovementRef | A pointer to the camera pawn’s movement component |
| Private | FTimerHandle | DelayStartTimerHandle | A timer handle for addressing issues of loading into the map during replication |
| Public | UMainUIBase\* | MainUI | A reference to the main UMG widget displayed during runtime of the game |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ARTSPlayerControllerBase() | Default constructor. Only enables tick and pause related events. |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Protected | Void | BeginPlay() | Shows mouse cursor and currently sets timer for delayed start. |
| Public | Void | Tick(float DeltaTime) | Runs edge scroll events |
| Public | Void | SetupInputComponent() | Bind functionality to input (do not edit unless adding in new user inputs) |
| Public | Void | DelayedBeginPlay() | Gets the user’s camera pawn (CameraPawnRef) and sends a reference to self to the server (stored in GameState). Clears and invalidates timer hammer |
| Public | Void | SendIdentityToServer(APlayerController\* SelfRef) | NOT USED (need to delete?) |
| Public | Void | CallMoveFoward(float Value) | What happens when W/S is pressed. This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallMoveRight(float Value) | What happens when A/D is pressed. This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallIncraseSpeedModifier() | What happens when shift is pressed, increasing movement speed modifier to 2. This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallDecreaseSpeedModifier() | What happens when shift is released, increasing movement speed modifier to 1. This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallZoomIn() | What happens when the mouse wheel up is triggered (zoom in). This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallZoomOut() | What happens when the mouse wheel down is triggered (zoom out). This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallZoomReset() | What happens when the input for zoom reset is triggered. The zoom returns to default. This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallPanReset() | What happens when the input for pan reset is triggered. The rotation returns to default. This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | CallCameraReset() | Returns zoom/pan both to default. This method has no actual effect, it just calls up the relevant method in the Camera Movement Component. |
| Public | Void | RoC\_PassDateTimeStruct(FDateTime TimeStruct) | Replicated on Client event that passes the DateTime structure to the UMG. Triggered GameState using by Interface event (SetGameDateTime). Triggers same event in MainUI. |
| Public | Void | RoC\_PassDayOfWeek(int GameDay) | Replicated on Client event that passes the Day of the Week to the UMG. Triggered GameState using by Interface event (SetGameDay). Triggers same event in MainUI. |
| Public | Void | RoC\_PassGameSpeed(float GameSpeed) | Replicated on Client event that passes Game Speed to the UMG. Triggered GameState using by Interface event (UpdateGameSpeed). Triggers same event in MainUI. |
| Public | Void | RoS\_SetGameSpeed(float GameSpeed) | Replicated on Server event that passes (and then sets) the GameSpeed from the UMG to the GameState. Triggered by UMG using interface event (UpdateGameSpeed). Triggers the same event in GameState. |

## Interface Related Functions

Remember these functions are defined with ‘\_Implementation’ and called with ‘Execute\_’

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Public | Bool | SetGameDateTime(FDateTime TimeStruct) | Used as an intermediate communication point between game state and UMG for passing the DateTime Structure to UMG. Triggers ROC\_passDateTimeStruct. |
| Public | Bool | SetGameDay(int GameDay) | Used as an intermediate communication point between game state and UMG for passing the Day of the Week to UMG. Triggers RoC\_PassDayOfWeek. |
| Public | Void | UpdateGameSpeed(float SpeedMultiplier) | Used as an intermediate communication point between game state and UMG for passing/Setting the GameSpeed in GameState. Triggers RoS\_SetGameSpeed. |
| Public | Bool | SetGameSpeedInt(float GameSpeed) | Used as an intermediate communication point between game state and UMG for passing GameSpeed to UMG. Triggers RoC\_PassGameSpeed. |

# RTSGameInstance

## References

|  |  |
| --- | --- |
| Header | Core/Settings/RTSGameInstance.h |
| Source | Private/Core/Settings/ RTSGameInstance.cpp |

## Remarks

This class is used to store all the controls for hosting, joining, leaving, etc. a session. Implementation of the Online Subsystems will occur here. **NOTE: At the time of starting this document (4 December 2021) this class is not completed. This likely will the LAST class completed.** Proposed cycle for this class: Update to the NULL online subsystem during dev cycle. Make all primary gameplay tests over VPN style devices (e.g., Hamachi). Once final changes are made, then this will be updated to the Steam system.

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
|  |  |  |  |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | URTSGameInstance(const FObjectInitializer& ObjectInitializer) | Sets the default values |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Public | Void | Init() | **Currently empty** |
| Public | Void | Host | **Currently** allows a player to host the game and server travel to the map.  **Currently a CVAR** (may leave as CVAR) |
| Public | Void | Join(const FString& Address) | **Currently** Takes an IP address to allow for players to join a listen server. Uses client travel to the map.  **Currently a CVAR** (may leave as CVAR) |

# RTSGameState

## References

|  |  |
| --- | --- |
| Header | Core/Settings/RTSGameState.h |
| Source | Private/Core/Settings/RTSGameState.cpp |

## Remarks

This class is the persistent class used to control the main game environment. This class contains the information above the host, active players, game time.

## Game Speed Breakdown

The table below should be used to determine what game speed is most desired. The examples given below are just to help understand the ratios for real time elapsed to time in game. For example, if you want 1 real world second to be 1 hour in game, then the GameSpeed should be 1. If you want 1 real hour to be 24 hours in game, then GameSpeed Should be 150.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Real time elapsed** | : | **GameSpeed** | = | **Time in Game** |
| 1 second | : | 1 | **=** | 1 game hour |
| 1 second | : | 60 | **=** | 1 game minute |
| 1 second | : | 3600 | **=** | 1 game second (real time) |
| 1 minute | : | 2.5 | **=** | 1 game day |
| 1 hour | : | 150 | **=** | 1 game day |

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| Protected | TArray<ARTSPlayerControllerBase\*> | PlayerRefs | Stores pointers to the active player controller in the session.  **NOTE TO SELF:** why note go up one level in the inheritance hierarchy? You are using interfaces after all… |
| Protected | ARTSPlayerControllerBase \* | HostPlayerControllerRef | A pointer to the host’s player controller |
| Protected | AActor\* | SunActorRef | A reference to the directional light used for the day/night cycle. |
| Protected | Int | Years | The year of the game (default 1100) |
| Protected | Int | Months | Game Month (default 10/October) |
| Protected | Int | Days | Day of the Month (default 21st) |
| Protected | Int | DayOfWeek | A numeric representation of what day of the week it is (range 1 to 7, where 1 = Sunday and 7 = Saturday) |
| Protected | Int | DayCounter | An int set to either 0 or 1. At 1, this variable triggers an update to Days and DayOfWeek. (potentially month and year as well, if changing months or year) |
| Protected | Int | Hours | Current hour in the game world (default is 13). This runs on 24hour clock. |
| Protected | Int | Minutes | Current minute in the game world (default 30). |
| Protected | Int | Seconds | Current Second in the game world (default 12). |
| Protected | Float | GameSpeed | How fast the game time moves. Default is 60. Break down is provided in remarks. |
| Protected | Float | DefaultGameSpeed | Should always be equal to GameSpeed (used for resetting Gamespeed) |
| Protected | FDateTime | DateTimeStruct | The structure that sores the game’s Date and Time data |
| Private | FTimerHandle | BeginPlayTimerHandle | A timer handled used for a delay. **NOTE:** This likely will be deprecated. |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ARTSGameState () | Default constructor. Sets up the GameTime, DateTimeStruct, and confirms actors should tick. |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Protected | Void | BeginPlay() | Server gets actor that controls day/night cycle and triggers timer (likely to be replaced) to collect player data. |
| Protected | Void | Tick(float DeltaTime) | Server runs the game clock and passes the game speed to the UMG (through the player controller). |
| Protected | Void | RoS\_DelayedCollectPlayerData() | Server collects information on all of the player controllers and determines who the host is. Once done it kills the timer. **Note:** This likely will be deprecated and the functions moved elsewhere. |
| Protected | Void | RoS\_SetAllPlayerControllerRefs() | Server calls this during DelayedCollectPlayerData (again likely to be replaced). It gets all the player controllers and stores the data. |
| Protected | Void | RoS\_CalculateTime(float DeltaTime, float CurrentGameSpeed, float GameTimeIn) | Server updates GameTime based on the tick (DeltaTime) and CurrentGameSpeed. This is the calculation that controls the length of the day (default 24 hours). Also checks if GameTime has done a 24 hour cycle (if it is has DayCounter goes from 0 to 1). |
| Protected | Void | RoS\_SetClockCalendar() | Takes Gametime and converts this float value into seconds, minutes, and hours. If Day Counter is 1, then Day of Week and Days will increment. If either go over their max values (7 and DaysInMonth(), respectively) they are reset to 1. Likewise month and year are updated. |
| Protected | Void | RoS\_SetDateTime(const int& Year, const int& Month, const int& Day, const int& Hour, const int& Minute, const int& Second) | The server event that stores the DateTimeStruct (called by RoS\_SetClockCalendar()). |
| Protected | Void | RoS\_PassDateTimeStruct() | Replicated on Server event that passes the DateTime structure to the UMG. Called on the tick. This data is passed to the Player Controller, which using the same interface function passes the information to the UMG. |
| Protected | Void | RoS\_PassDayOfWeek() | Replicated on Server event that passes the day of the week value to the UMG. Called on the tick. This data is passed to the Player Controller, which using the same interface function passes the information to the UMG. |
| Protected | Void | RoS\_SetGameSpeed(const float& SpeedMultiplier) | Replicated on Server event sets the GameSpeed. Uses the DefaultGameSpeed value as a constant base. It takes this value and multiplies it by a value set in the UMG. |
| Protected | Void | RoS\_PassGameSpeed() | Replicated on Server event that passes the GameSpeedValue to those actors (e.g., MainUI UMG) that need the value. Called on the tick. |
| Protected | Void | RoS\_GetSunRotationActor() | The function that actually stores the Day/Night cycle actor. Server locates the actor and stores it. |
| Public | Float | GetGameTime() | Returns GameTime |

## Interface Related Functions

Remember these functions are defined with ‘\_Implementation’ and called with ‘Execute\_’

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| Public | Void | UpdateGameSpeed(float SpeedMultiplier) | The interface function that allows players to change the gamespeed. This function is triggered (typically) by the player via the UMG, to the controller, then finally here to GameState. This passes the data from the user to the server and calls RoS\_SetGameSpeed (logging optional to see if it is was server/host, or player) |
| Public | Bool | AddPlayerController(APlayerController\* PlayerControllerRef) | A backup method for a player to be added to the list of PlayerControllers (may be useful for resolving disconnects and reconnects). Currently I do not *believe* this is used. Corresponding event (that triggers this interface) is in the RTSPlayerControllerBase. |

# UnitMasterBase

## References

|  |  |
| --- | --- |
| Header |  |
| Source |  |

## Remarks

Lorem Ipsum

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ABuildingMasterBase () | Sets the default values |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

# GameClockWidget

## References

|  |  |
| --- | --- |
| Header |  |
| Source |  |

## Remarks

Lorem Ipsum

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ABuildingMasterBase () | Sets the default values |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

## Interface Related Functions

Remember these functions are defined with ‘\_Implementation’ and called with ‘Execute\_’

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |

# MainUIBase

## References

|  |  |
| --- | --- |
| Header |  |
| Source |  |

## Remarks

Lorem Ipsum

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Type** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

## Constructors

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Public | ABuildingMasterBase () | Sets the default values |

## Functions

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |
|  |  |  |  |

## Interface Related Functions

Remember these functions are defined with ‘\_Implementation’ and called with ‘Execute\_’

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Return** | **Name** | **Description** |
| - | - | - | - |