OHL Technical Document

# Character System

Overview - There is Entity Class that has basic elements that character and enemy needs. This Character system is inheritance with entity class. Character Manager has all the information even enemies. On changing the scene, we can get the attributes that saved in the Character Manager (global). So, if you want to get some information about character or enemy, call the Get function in Character Manager. However, for this, whenever change the value in the character, we have to save the data into Character Manager using Set function. Character Setup is register characters and it will match what player selected the characters. Furthermore, Party System will remember the order that the player selected character. In each scene, we can get the order and saved attribute values.

# How to use

First, we need CharacterManager, you have to create an Empty object and put our CharacterManager.cs in Script. Add size in the player section and put the prefabs name of “Character” into the Elements. Second, create a new object for players in the Scene. In the player, you have to add script component “Character.cs”. If you want to make a button, use this player object. Please make sure the character ID, if you want to make a player more than one.

# CODING METHODS

Entity.cs

Purpose : Entity class have basic elements that what they need both character and enemy for dictionary.

|  |  |  |
| --- | --- | --- |
| Type | Value Name | Description |
| **Int** | ID | Each character and enemy have a unique ID |
| **Float** | HP | HP |
| **Attributes** | attribute | There are 5 main attributes (con, wis, str, dex, int) in “Attributes” class |
| **GameObject** | target | Each character and enemy need to know who they are attacking |
|  |  |  |

Character.cs

Purpose : Character have Level and Attributes.

|  |  |  |  |
| --- | --- | --- | --- |
| Part | Function Name | Description |  |
| **Level** | void Levelup()  void AddExp() | If level value greater than experience require, it will be level up or you can use AddExp() function that is add 100 for test. |  |
| **Attributes** | // Attribute Point  void AddAttributePoint()  //Con  void AddMaxHealth()  void AddArmor ()  void AddElementalWard()  void AddAfflictionResistance()  //Wis  void AddMaxResource()  void AddResourceRegeneration()  void AddResourceCosReduction()  //STR  void AddArmorBreak()  void AddPhysicalDamage()  void AddSovereignDamage ()  //DEX  void AddCriticalChance()  void AddCriticalDamage()  void AddAttackSpeed()  //INT  void AddElemetalDamage()  void AddWardBreak()  void AddCastingSpeed() | \*Attribute point is amount that the player can add the attribute values  Add one | Customization options.  The 3 main stats (STR, DEX, INT) have substats so that attributes look simple at a surface glance.  \*Note: Consider adding substats for CON and WIS. |
| **Attribute Distribution** | Public AttributeDistribution AD | This value contain how many get that attribute point. |  |
| **Name** | Public string Name | Unique name |  |
| **Level** | Public int Level | Level |  |
| **Experience** | Public int XP  Public int XP\_required | Experience and require value for level up |  |

CharacterManager.cs

Purpose

Register characters

Save and give our Character attributes value during the game.

|  |  |  |
| --- | --- | --- |
| Part | Function Name | Description |
| **Register** | public static void RegisterCharacter(string \_ID, Character \_character)  public static void UnRegisterCharacter(string \_characterID) | Input the Character name and character class for register into Character list |
| **Gettter (Character)** | public static Character GetCharacter(int \_characterID) | For get the character object anywhere |
| **Getter**  **(Attributes)** | int GetAttributePoint(int id)  int GetLevel(int id)  int GetExp(int id)  int GetExpRequried(int id)  //Con  float GetHP(int id)  float GetMaxHealth(int id)  float GetArmor(int id)  float GetElementalWard(int id)  float GetAfflictionResistance(int id)  //Wis  float GetMaxResource(int id)  float GetResourceRegeneration(int id)  float GetResourceCosReduction(int id)  //STR  float GetArmorBreak(int id)  float GetPhysicalDamage(int id)  float GetSovereignDamage(int id)  //DEX  float GetCriticalChance(int id)  float GetCriticalDamage(int id)  float GetAttackSpeed(int id)  //INT  float GetElemetalDamage(int id)  float GetWardBreak(int id)  float GetCastingSpeed(int id) | For getting our Attribute values  \*if you have made just one character, then the id is 0. |
| **Setter** | //Attribute Point  Void SetAttributePoint(int id, int val)  Void SetLevel(int id, int val)  Void SetExp(int id, int val)  Void SetExpRequired(int id, int val)  //Con  void SetMaxHealth(int id, float value)  void SetArmor(int id, float value)  void SetElementalWard(int id, float value)  void SetAfflictionResistance(int id, float value)  //Wis  void SetMaxResource(int id, float value)  void SetResourceRegeneration(int id, float value)  void SetResourceCosReduction(int id, float value)  //STR  void SetArmorBreak(int id, float value)  void SetPhysicalDamage(int id, float value)  void SetSovereignDamage(int id, float value)  //DEX  void SetCriticalChance(int id, float value)  void SetCriticalDamage(int id, float value)  void SetAttackSpeed(int id, float value)  //INT  void SetElemetalDamage(int id, float value)  void SetWardBreak(int id, float value)  void SetCastingSpeed(int id, float value) | For setting the attributes | . |

CharacterSetup.cs

Purpose : Register characters

|  |  |  |
| --- | --- | --- |
| Part | Function Name | Description |
| **Register** | Private void OnEnable()  Private void OnDisable() | Enroll the character with object name. |

Attributes.cs

Purpose : Manage the percentage of attributes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute | | Description | Possible Substat Points | Maximum Value |
| **Constitution** | | **Base: Gives flat hp based on level** |  |  |
| Max Health | +10% Maximum HP | 20 | 200% |
| Armor | +5% Armor | 5 | 25% |
| Elemental Warding | +5% Elemental Warding | 5 | 25% |
| Affliction Resistance | +10% affliction resistance | 5 | 50% |
| **Wisdom** | | **Base: Gives flat resource based on level** |  |  |
| Max Resource | +10% Maximum Resource | 20 | 200% |
| Resource Regeneration | +20% Resource Regeneration | 20 | 400% |
| Resource Cost Reduction | -10% Resource Cost | 5 | -50% |
| **Strength** | | **Base: Gives +10% Physical Damage per point** | **20** | **200%** |
| Armor Break | +5% Armor Break | 20 | 100% |
| Physical Penetration | +5% Physical Penetration | 16 | 80% |
| Sovereign Damage | +10% Physical Damage | 20 | 200% **+ 200%** |
| **Dexterity** | | **Base: Gives +10% Critical Damage per point** | **20** | **200%** |
| Critical Chance | +5% Critical Chance | 20 | 80% |
| Attack Speed | +5% Attack Speed | 10 | 50% |
| Critical Damage | +10% Critical Damage | 20 | 200% **+ 200%** |
| **Intelligence** | | **Base: Gives +10% Elemental Damage per point** | **20** | **200%** |
| Casting Speed | +5% Casting Speed | 10 | 50% |
| Ward Break | +5% Elemental Penetration | 16 | 80% |
| Casting Speed | +10% Casting Speed | 20 | 200% **+ 200%** |

AttributeDistribution.cs

Purpose : Calculate current attribute point

|  |  |  |
| --- | --- | --- |
| Part | Function Name | Description |
| **Getter** | //Con  Float AD\_MaxHealth(float init)  Float AD\_Armor(float init)  Float AD\_ElementalWard(float init)  Float AD\_AfflictionResistance(float init)  //Wis  Float AD\_MaxResource(float init)  Float AD\_ResourceRegeneration(float init)  Float AD\_ResourceCostReduction(float init)  //Str  Float AD\_ArmorBreak(float init)  Float AD\_PhysicalDamage(float init)  Float AD\_SovereignDamage(float init)  //Dex  Float AD\_CriticalChance(float init)  Float AD\_CriticalDamage(float init)  Float AD\_AttackSpeed(float init)  //Int  Float AD\_ElemetalDamage(float init)  Float AD\_WardBreak(float init)  Float AD\_CastingSpeed(float init) | It will be return the character’s attribute value that calculated with attribute points.  (Init + (Init\* Attributes\*Attributes\_Point))  \*Init – initial value what we set each character.  \*Attributes – attributes value what we set the percentage.  \*Attributes\_Point – how many point added that attributes. |