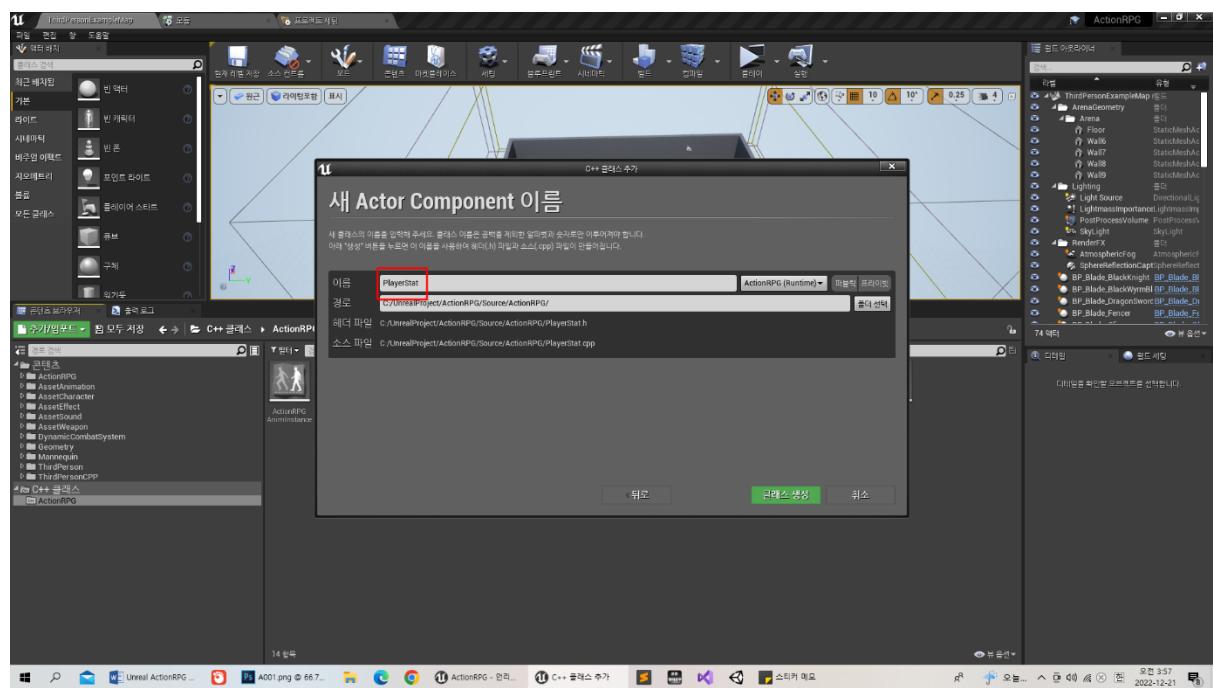
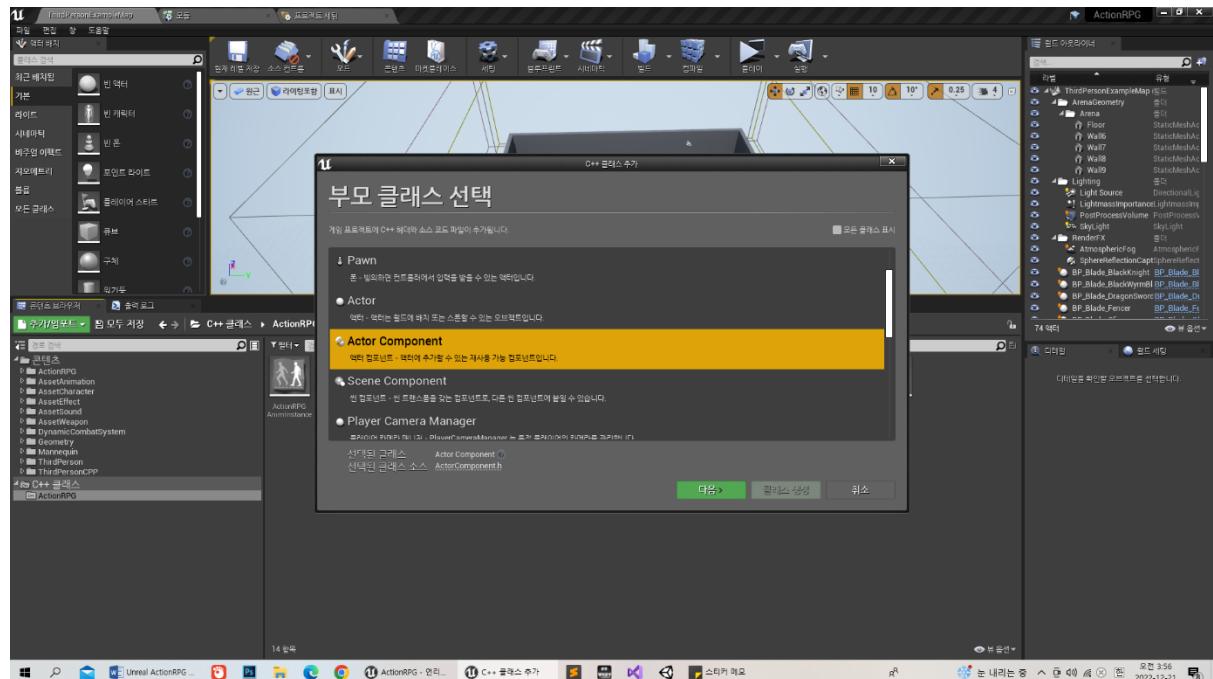
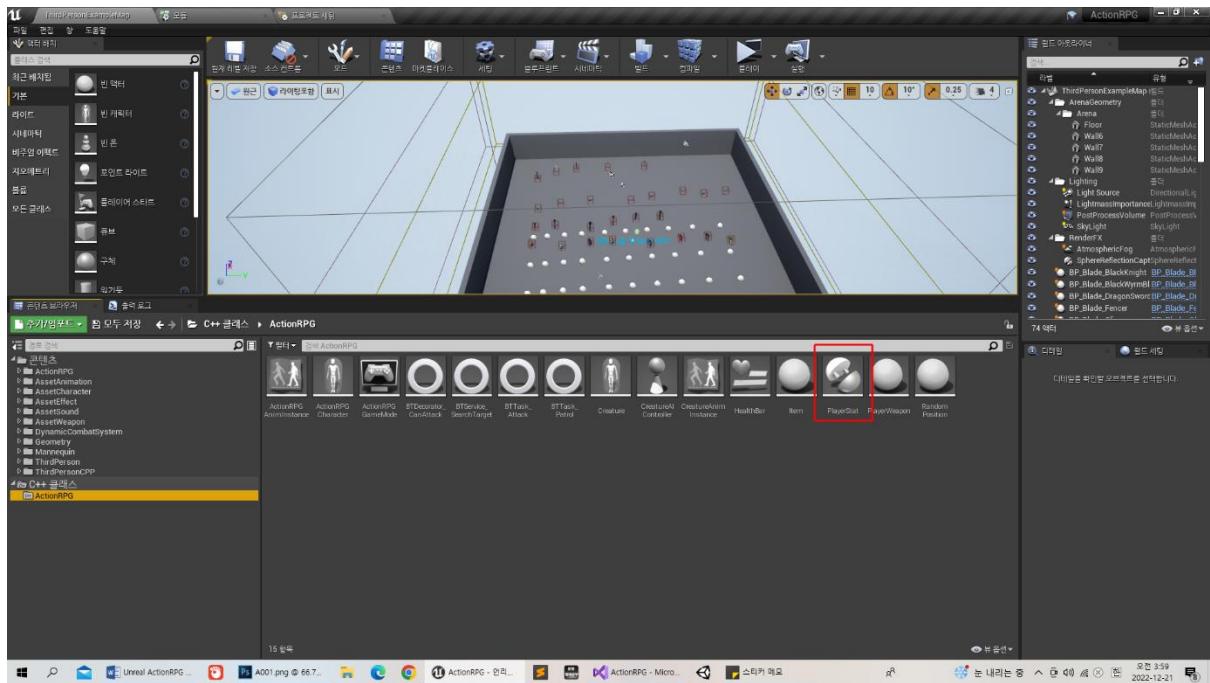


수치값을 코드에 넣는 것을 하드코딩이라고 합니다. 하드코딩을 없애주도록 합니다.

플레이어부터 해 주도록 합니다.

ActorComponent클래스를 부모 클래스로 상속받는 PlayerStat이라는 이름의 클래스를 정의해 줍니다.

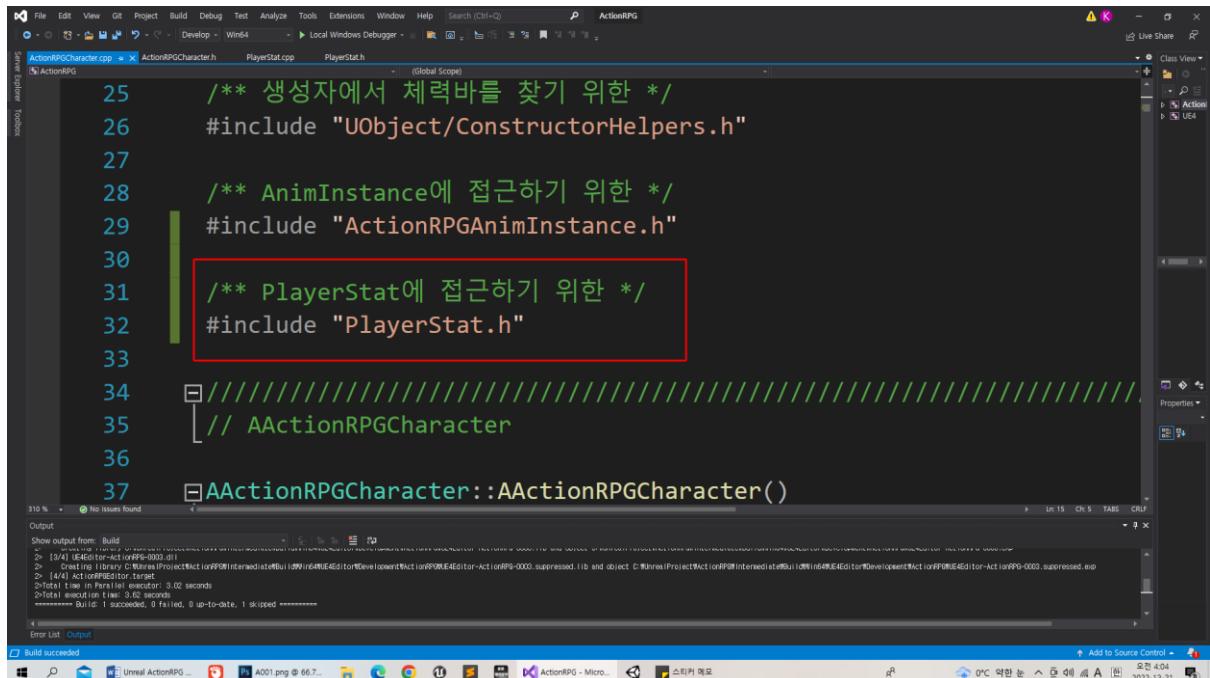




플레이어 클래스에서 생성해 주도록 합니다.

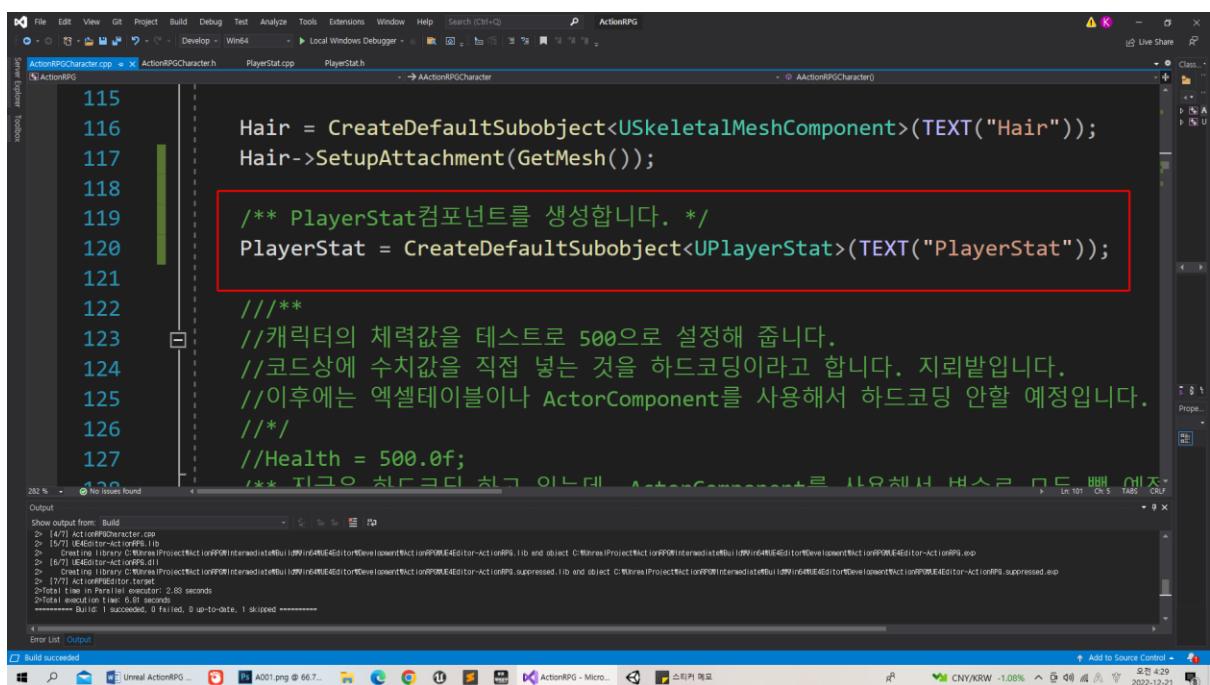
```

File Edit View G Engine - Develop - Win64 Local Windows Debugger
PlayerWeapon.cpp ActionRPGCharacter.cpp ActionRPGCharacter.h PlayerStat.cpp PlayerStat.h
  ActionRPGCharacter
  207
  208
  209
  210
  211  /** PlayerStat을 저장할 변수를 선언합니다. */
  212  UPROPERTY(EditDefaultsOnly, Category = "PlayerStat", meta = (AllowPrivateAccess = true))
  213  class UPlayerStat* PlayerStat;
  214
  215
  216
  217
  218
  219
  220
  221
  222
  223
  224
  225
  226
  227
  228
  229
  230
  231
  232
  233
  234
  235
  236
  237
  238
  239
  240
  241
  242
  243
  244
  245
  246
  247
  248
  249
  250
  251
  252
  253
  254
  255
  256
  257
  258
  259
  260
  261
  262
  263
  264
  265
  266
  267
  268
  269
  270
  271
  272
  273
  274
  275
  276
  277
  278
  279
  280
  281
  282
  283
  284
  285
  286
  287
  288
  289
  290
  291
  292
  293
  294
  295
  296
  297
  298
  299
  300
  301
  302
  303
  304
  305
  306
  307
  308
  309
  310
  311
  312
  313
  314
  315
  316
  317
  318
  319
  320
  321
  322
  323
  324
  325
  326
  327
  328
  329
  330
  331
  332
  333
  334
  335
  336
  337
  338
  339
  340
  341
  342
  343
  344
  345
  346
  347
  348
  349
  350
  351
  352
  353
  354
  355
  356
  357
  358
  359
  360
  361
  362
  363
  364
  365
  366
  367
  368
  369
  370
  371
  372
  373
  374
  375
  376
  377
  378
  379
  380
  381
  382
  383
  384
  385
  386
  387
  388
  389
  390
  391
  392
  393
  394
  395
  396
  397
  398
  399
  400
  401
  402
  403
  404
  405
  406
  407
  408
  409
  410
  411
  412
  413
  414
  415
  416
  417
  418
  419
  420
  421
  422
  423
  424
  425
  426
  427
  428
  429
  430
  431
  432
  433
  434
  435
  436
  437
  438
  439
  440
  441
  442
  443
  444
  445
  446
  447
  448
  449
  450
  451
  452
  453
  454
  455
  456
  457
  458
  459
  460
  461
  462
  463
  464
  465
  466
  467
  468
  469
  470
  471
  472
  473
  474
  475
  476
  477
  478
  479
  480
  481
  482
  483
  484
  485
  486
  487
  488
  489
  490
  491
  492
  493
  494
  495
  496
  497
  498
  499
  500
  501
  502
  503
  504
  505
  506
  507
  508
  509
  510
  511
  512
  513
  514
  515
  516
  517
  518
  519
  520
  521
  522
  523
  524
  525
  526
  527
  528
  529
  530
  531
  532
  533
  534
  535
  536
  537
  538
  539
  540
  541
  542
  543
  544
  545
  546
  547
  548
  549
  550
  551
  552
  553
  554
  555
  556
  557
  558
  559
  560
  561
  562
  563
  564
  565
  566
  567
  568
  569
  570
  571
  572
  573
  574
  575
  576
  577
  578
  579
  580
  581
  582
  583
  584
  585
  586
  587
  588
  589
  590
  591
  592
  593
  594
  595
  596
  597
  598
  599
  600
  601
  602
  603
  604
  605
  606
  607
  608
  609
  610
  611
  612
  613
  614
  615
  616
  617
  618
  619
  620
  621
  622
  623
  624
  625
  626
  627
  628
  629
  630
  631
  632
  633
  634
  635
  636
  637
  638
  639
  640
  641
  642
  643
  644
  645
  646
  647
  648
  649
  650
  651
  652
  653
  654
  655
  656
  657
  658
  659
  660
  661
  662
  663
  664
  665
  666
  667
  668
  669
  670
  671
  672
  673
  674
  675
  676
  677
  678
  679
  680
  681
  682
  683
  684
  685
  686
  687
  688
  689
  690
  691
  692
  693
  694
  695
  696
  697
  698
  699
  700
  701
  702
  703
  704
  705
  706
  707
  708
  709
  710
  711
  712
  713
  714
  715
  716
  717
  718
  719
  720
  721
  722
  723
  724
  725
  726
  727
  728
  729
  730
  731
  732
  733
  734
  735
  736
  737
  738
  739
  740
  741
  742
  743
  744
  745
  746
  747
  748
  749
  750
  751
  752
  753
  754
  755
  756
  757
  758
  759
  760
  761
  762
  763
  764
  765
  766
  767
  768
  769
  770
  771
  772
  773
  774
  775
  776
  777
  778
  779
  780
  781
  782
  783
  784
  785
  786
  787
  788
  789
  790
  791
  792
  793
  794
  795
  796
  797
  798
  799
  800
  801
  802
  803
  804
  805
  806
  807
  808
  809
  810
  811
  812
  813
  814
  815
  816
  817
  818
  819
  820
  821
  822
  823
  824
  825
  826
  827
  828
  829
  830
  831
  832
  833
  834
  835
  836
  837
  838
  839
  840
  841
  842
  843
  844
  845
  846
  847
  848
  849
  850
  851
  852
  853
  854
  855
  856
  857
  858
  859
  860
  861
  862
  863
  864
  865
  866
  867
  868
  869
  870
  871
  872
  873
  874
  875
  876
  877
  878
  879
  880
  881
  882
  883
  884
  885
  886
  887
  888
  889
  890
  891
  892
  893
  894
  895
  896
  897
  898
  899
  900
  901
  902
  903
  904
  905
  906
  907
  908
  909
  910
  911
  912
  913
  914
  915
  916
  917
  918
  919
  920
  921
  922
  923
  924
  925
  926
  927
  928
  929
  930
  931
  932
  933
  934
  935
  936
  937
  938
  939
  940
  941
  942
  943
  944
  945
  946
  947
  948
  949
  950
  951
  952
  953
  954
  955
  956
  957
  958
  959
  960
  961
  962
  963
  964
  965
  966
  967
  968
  969
  970
  971
  972
  973
  974
  975
  976
  977
  978
  979
  980
  981
  982
  983
  984
  985
  986
  987
  988
  989
  990
  991
  992
  993
  994
  995
  996
  997
  998
  999
  1000
  1001
  1002
  1003
  1004
  1005
  1006
  1007
  1008
  1009
  1010
  1011
  1012
  1013
  1014
  1015
  1016
  1017
  1018
  1019
  1020
  1021
  1022
  1023
  1024
  1025
  1026
  1027
  1028
  1029
  1030
  1031
  1032
  1033
  1034
  1035
  1036
  1037
  1038
  1039
  1040
  1041
  1042
  1043
  1044
  1045
  1046
  1047
  1048
  1049
  1050
  1051
  1052
  1053
  1054
  1055
  1056
  1057
  1058
  1059
  1060
  1061
  1062
  1063
  1064
  1065
  1066
  1067
  1068
  1069
  1070
  1071
  1072
  1073
  1074
  1075
  1076
  1077
  1078
  1079
  1080
  1081
  1082
  1083
  1084
  1085
  1086
  1087
  1088
  1089
  1090
  1091
  1092
  1093
  1094
  1095
  1096
  1097
  1098
  1099
  1100
  1101
  1102
  1103
  1104
  1105
  1106
  1107
  1108
  1109
  1110
  1111
  1112
  1113
  1114
  1115
  1116
  1117
  1118
  1119
  1120
  1121
  1122
  1123
  1124
  1125
  1126
  1127
  1128
  1129
  1130
  1131
  1132
  1133
  1134
  1135
  1136
  1137
  1138
  1139
  1140
  1141
  1142
  1143
  1144
  1145
  1146
  1147
  1148
  1149
  1150
  1151
  1152
  1153
  1154
  1155
  1156
  1157
  1158
  1159
  1160
  1161
  1162
  1163
  1164
  1165
  1166
  1167
  1168
  1169
  1170
  1171
  1172
  1173
  1174
  1175
  1176
  1177
  1178
  1179
  1180
  1181
  1182
  1183
  1184
  1185
  1186
  1187
  1188
  1189
  1190
  1191
  1192
  1193
  1194
  1195
  1196
  1197
  1198
  1199
  1200
  1201
  1202
  1203
  1204
  1205
  1206
  1207
  1208
  1209
  1210
  1211
  1212
  1213
  1214
  1215
  1216
  1217
  1218
  1219
  1220
  1221
  1222
  1223
  1224
  1225
  1226
  1227
  1228
  1229
  1230
  1231
  1232
  1233
  1234
  1235
  1236
  1237
  1238
  1239
  1240
  1241
  1242
  1243
  1244
  1245
  1246
  1247
  1248
  1249
  1250
  1251
  1252
  1253
  1254
  1255
  1256
  1257
  1258
  1259
  1260
  1261
  1262
  1263
  1264
  1265
  1266
  1267
  1268
  1269
  1270
  1271
  1272
  1273
  1274
  1275
  1276
  1277
  1278
  1279
  1280
  1281
  1282
  1283
  1284
  1285
  1286
  1287
  1288
  1289
  1290
  1291
  1292
  1293
  1294
  1295
  1296
  1297
  1298
  1299
  1300
  1301
  1302
  1303
  1304
  1305
  1306
  1307
  1308
  1309
  1310
  1311
  1312
  1313
  1314
  1315
  1316
  1317
  1318
  1319
  1320
  1321
  1322
  1323
  1324
  1325
  1326
  1327
  1328
  1329
  1330
  1331
  1332
  1333
  1334
  1335
  1336
  1337
  1338
  1339
  1340
  1341
  1342
  1343
  1344
  1345
  1346
  1347
  1348
  1349
  1350
  1351
  1352
  1353
  1354
  1355
  1356
  1357
  1358
  1359
  1360
  1361
  1362
  1363
  1364
  1365
  1366
  1367
  1368
  1369
  1370
  1371
  1372
  1373
  1374
  1375
  1376
  1377
  1378
  1379
  1380
  1381
  1382
  1383
  1384
  1385
  1386
  1387
  1388
  1389
  1390
  1391
  1392
  1393
  1394
  1395
  1396
  1397
  1398
  1399
  1400
  1401
  1402
  1403
  1404
  1405
  1406
  1407
  1408
  1409
  1410
  1411
  1412
  1413
  1414
  1415
  1416
  1417
  1418
  1419
  1420
  1421
  1422
  1423
  1424
  1425
  1426
  1427
  1428
  1429
  1430
  1431
  1432
  1433
  1434
  1435
  1436
  1437
  1438
  1439
  1440
  1441
  1442
  1443
  1444
  1445
  1446
  1447
  1448
  1449
  1450
  1451
  1452
  1453
  1454
  1455
  1456
  1457
  1458
  1459
  1460
  1461
  1462
  1463
  1464
  1465
  1466
  1467
  1468
  1469
  1470
  1471
  1472
  1473
  1474
  1475
  1476
  1477
  1478
  1479
  1480
  1481
  1482
  1483
  1484
  1485
  1486
  1487
  1488
  1489
  1490
  1491
  1492
  1493
  1494
  1495
  1496
  1497
  1498
  1499
  1500
  1501
  1502
  1503
  1504
  1505
  1506
  1507
  1508
  1509
  1510
  1511
  1512
  1513
  1514
  1515
  1516
  1517
  1518
  1519
  1520
  1521
  1522
  1523
  1524
  1525
  1526
  1527
  1528
  1529
  1530
  1531
  1532
  1533
  1534
  1535
  1536
  1537
  1538
  1539
  1540
  1541
  1542
  1543
  1544
  1545
  1546
  1547
  1548
  1549
  1550
  1551
  1552
  1553
  1554
  1555
  1556
  1557
  1558
  1559
  1560
  1561
  1562
  1563
  1564
  1565
  1566
  1567
  1568
  1569
  1570
  1571
  1572
  1573
  1574
  1575
  1576
  1577
  1578
  1579
  1580
  1581
  1582
  1583
  1584
  1585
  1586
  1587
  1588
  1589
  1590
  1591
  1592
  1593
  1594
  1595
  1596
  1597
  1598
  1599
  1600
  1601
  1602
  1603
  1604
  1605
  1606
  1607
  1608
  1609
  1610
  1611
  1612
  1613
  1614
  1615
  1616
  1617
  1618
  1619
  1620
  1621
  1622
  1623
  1624
  1625
  1626
  1627
  1628
  1629
  1630
  1631
  1632
  1633
  1634
  1635
  1636
  1637
  1638
  1639
  1640
  1641
  1642
  1643
  1644
  1645
  1646
  1647
  1648
  1649
  1650
  1651
  1652
  1653
  1654
  1655
  1656
  1657
  1658
  1659
  1660
  1661
  1662
  1663
  1664
  1665
  1666
  1667
  1668
  1669
  1670
  1671
  1672
  1673
  1674
  1675
  1676
  1677
  1678
  1679
  1680
  1681
  1682
  1683
  1684
  1685
  1686
  1687
  1688
  1689
  1690
  1691
  1692
  1693
  1694
  1695
  1696
  1697
  1698
  1699
  1700
  1701
  1702
  1703
  1704
  1705
  1706
  1707
  1708
  1709
  1710
  1711
  1712
  1713
  1714
  1715
  1716
  1717
  1718
  1719
  1720
  1721
  1722
  1723
  1724
  1725
  1726
  1727
  1728
  1729
  1730
  1731
  1732
  1733
  1734
  1735
  1736
  1737
  1738
  1739
  1740
  1741
  1742
  1743
  1744
  1745
  1746
  1747
  1748
  1749
  1750
  1751
  1752
  1753
  1754
  1755
  1756
  1757
  1758
  1759
  1760
  1761
  1762
  1763
  1764
  1765
  1766
  1767
  1768
  1769
  1770
  1771
  1772
  1773
  1774
  1775
  1776
  1777
  1778
  1779
  1780
  1781
  1782
  1783
  1784
  1785
  1786
  1787
  1788
  1789
  1790
  1791
  1792
  1793
  1794
  1795
  1796
  1797
  1798
  1799
  1800
  1801
  1802
  1803
  1804
  1805
  1806
  1807
  1808
  1809
  1810
  1811
  1812
  1813
  1814
  1815
  1816
  1817
  1818
  1819
  1820
  1821
  1822
  1823
  1824
  1825
  1826
  1827
  1828
  1829
  1830
  1831
  1832
  1833
  1834
  1835
  1836
  1837
  1838
  1839
  1840
  1841
  1842
  1843
  1844
  1845
  1846
  1847
  1848
  1849
  1850
  1851
  1852
  1853
  1854
  1855
  1856
  1857
  1858
  1859
  1860
  1861
  1862
  1863
  1864
  1865
  1866
  1867
  1868
  1869
  1870
  1871
  1872
  1873
  1874
  1875
  1876
  1877
  1878
  
```



```
25     /** 생성자에서 체력바를 찾기 위한 */
26     #include "UObject/ConstructorHelpers.h"
27
28     /** AnimInstance에 접근하기 위한 */
29     #include "ActionRPGAnimInstance.h"
30
31     /** PlayerStat에 접근하기 위한 */
32     #include "PlayerStat.h"
33
34     //////////////////////////////////////////////////////////////////
35     // AActionRPGCharacter
36
37     AAActionRPGCharacter::AAActionRPGCharacter()
```

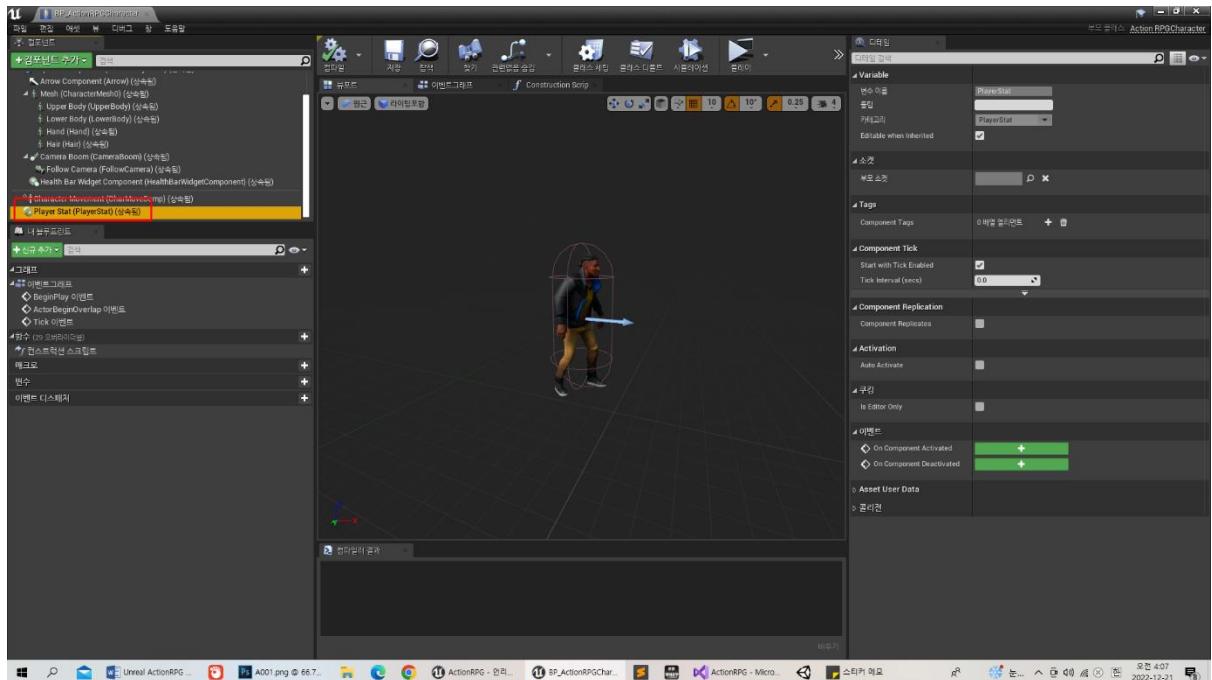
Output:
Build succeeded
2023-12-21 20:40:40



```
115
116
117     Hair = CreateDefaultSubobject<USkeletalMeshComponent>(TEXT("Hair"));
118     Hair->SetupAttachment(GetMesh());
119
120     /** PlayerStat컴포넌트를 생성합니다. */
121     PlayerStat = CreateDefaultSubobject<UPlayerStat>(TEXT("PlayerStat"));
122
123     /**
124     //캐릭터의 체력값을 테스트로 500으로 설정해 줍니다.
125     //코드상에 수치값을 직접 넣는 것을 하드코딩이라고 합니다. 지뢰밭입니다.
126     //이후에는 엑셀레이블이나 ActorComponent를 사용해서 하드코딩 안할 예정입니다.
127     //*/
128     //Health = 500.0f;
```

Output:
Build succeeded
2023-12-21 20:40:40

결과를 확인해 봅니다.

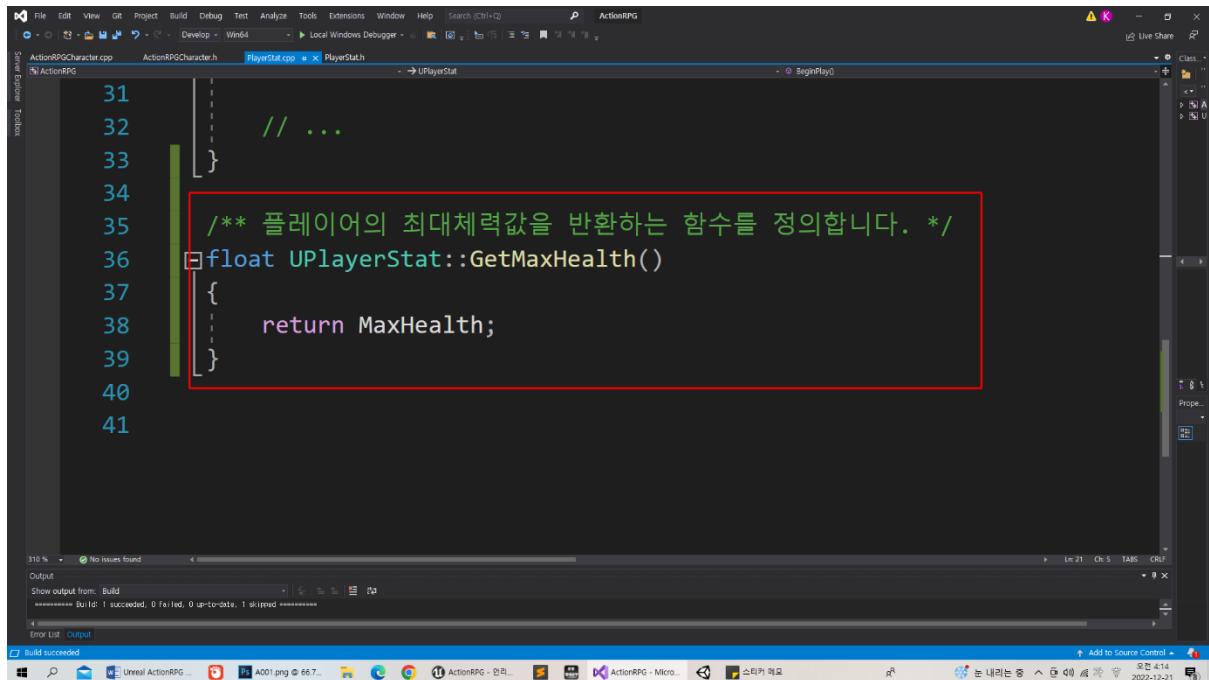


```

22
23     public:
24         // Called every frame
25         virtual void TickComponent(float DeltaTime, ELevelTick TickType, FActorComponentTickFunction* ThisTickFunction) override;
26
27     private:
28         /** 무기를 장착할 때 필요한 소켓이름은 저장할 변수를 선언합니다. */
29         UPROPERTY(EditAnywhere, Category = "PlayerStat", meta = (AllowPrivateAccess = "true"))
30         FName RightWeaponSocket;
31
32         /** 플레이어의 최대 체력값을 저장해 둘 변수를 선언합니다. */
33         UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Health", meta = (AllowPrivateAccess = "true"))
34         float MaxHealth;
35
36     public:
37         /** 플레이어의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
38         float GetMaxHealth();
39     };
40

```

The code snippet shows the implementation of the PlayerStat component in C++. It includes private variables for weapon sockets and health, and a public function to get the maximum health. The code is annotated with comments explaining the purpose of each section.

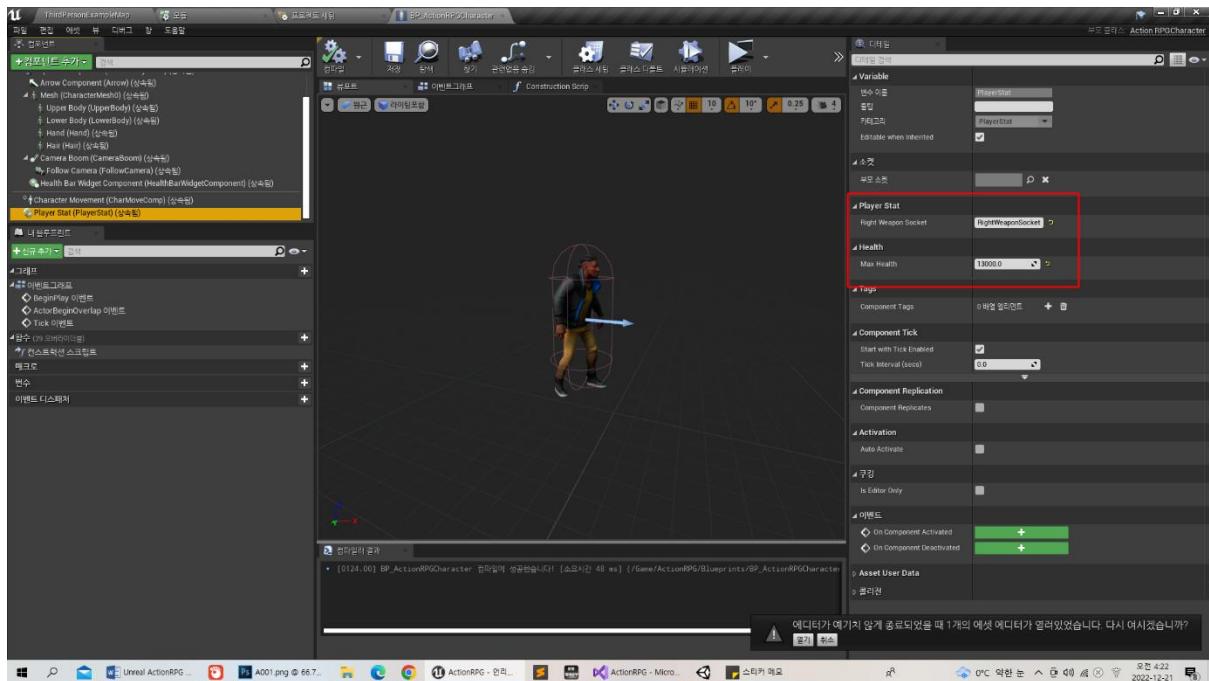


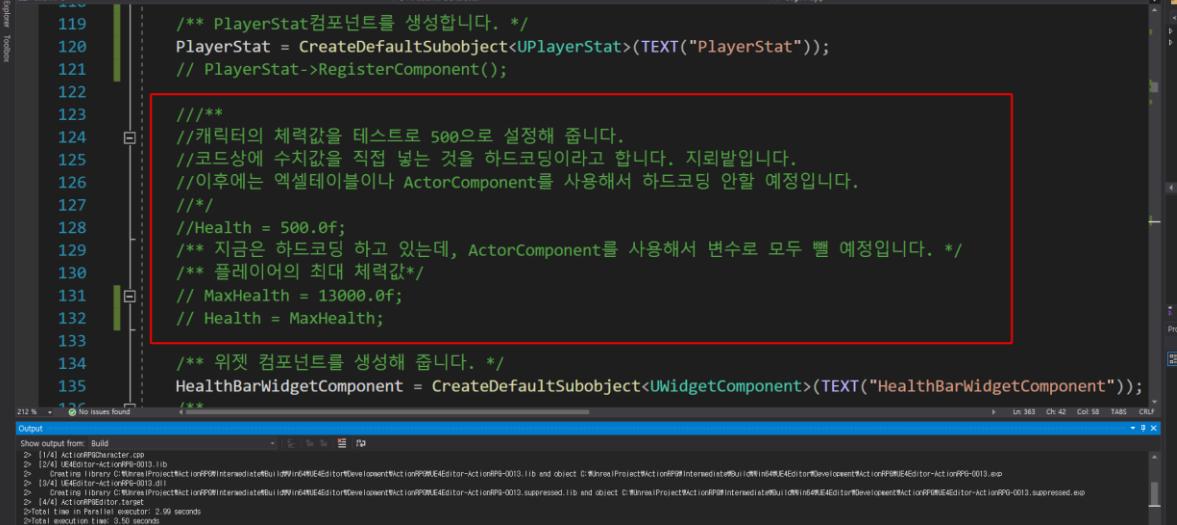
```

31 // ...
32 }
33
34
35 /** 플레이어의 최대체력값을 반환하는 함수를 정의합니다. */
36 float UPlayerStat::GetMaxHealth()
37 {
38     return MaxHealth;
39 }
40
41

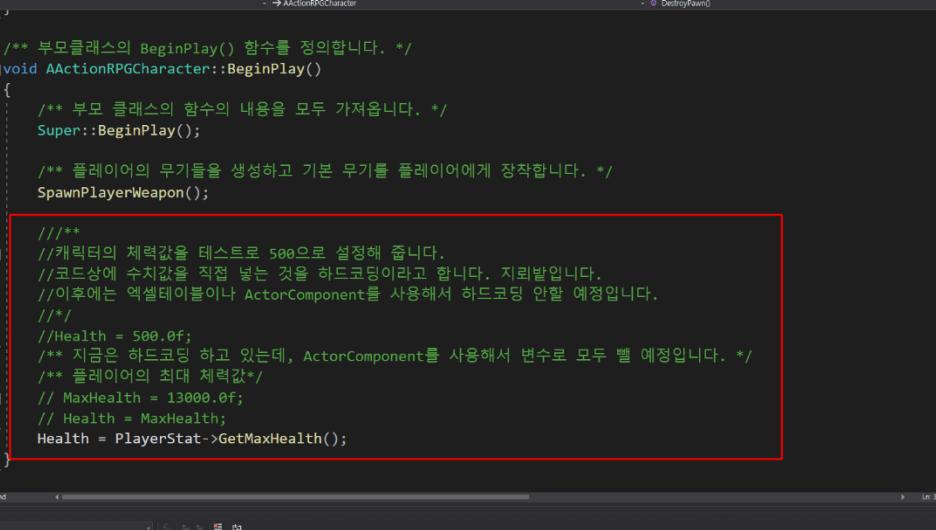
```

디테일 패널에서 확인해 봅니다.

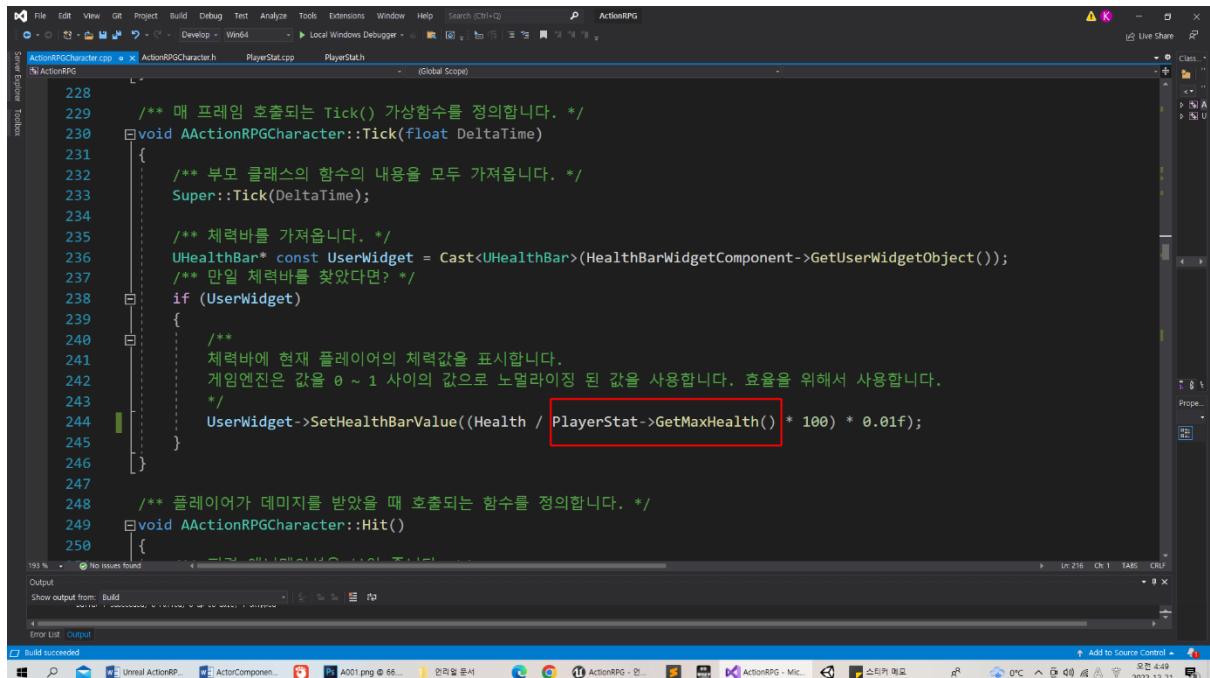




```
119     /** PlayerStat 컴포넌트를 생성합니다. */
120     PlayerStat = CreateDefaultSubobject<UPlayerStat>(TEXT("PlayerStat"));
121     // PlayerStat->RegisterComponent();
122
123     /**
124     //캐릭터의 체력값을 테스트로 500으로 설정해 줍니다.
125     //코드상에 수치값을 직접 넣는 것을 하드코딩이라고 합니다. 저희방입니다.
126     //이후에는 액셀레이터이나 ActorComponent를 사용해서 하드코딩 안할 예정입니다.
127     */
128
129     //Health = 500.0f;
130     /** 지금은 하드코딩 하고 있는데, ActorComponent를 사용해서 변수로 모두 뺄 예정입니다. */
131     /** 플레이어의 최대 체력값*/
132     // MaxHealth = 13000.0f;
133     // Health = MaxHealth;
134
135     /**
136     //위젯 컴포넌트를 생성해 줍니다.
137     HealthBarWidgetComponent = CreateDefaultSubobject<UWidgetComponent>(TEXT("HealthBarWidgetComponent"));
138
139     /**
140
141     Show output from: Build
142     2> [1/4] ActionRPGCharacter.cpp
143     2> [2/4] ActionRPGCharacter.h
144     2> [3/4] UE4Editor-ActionRPG-0013.dll
145     2> [4/4] ActionRPGCharacter
146     2>Total time in Parallel Executor: 2.99 seconds
147     2>Total execution time: 0.56 seconds
148     2>Build 1 succeeded, 0 failed, 0 up-to-date, 1 skipped
149
150 Error List - Output
151
152 Build succeeded
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
886
887
888
889
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
989
990
991
992
993
994
995
996
997
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2197
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2297
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2389
2390
2391
2392
2393
2394
2395
2396
2397
2397
2398
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2439
2440
2441
2442
2443
2444
2445
2446
2447
244
```

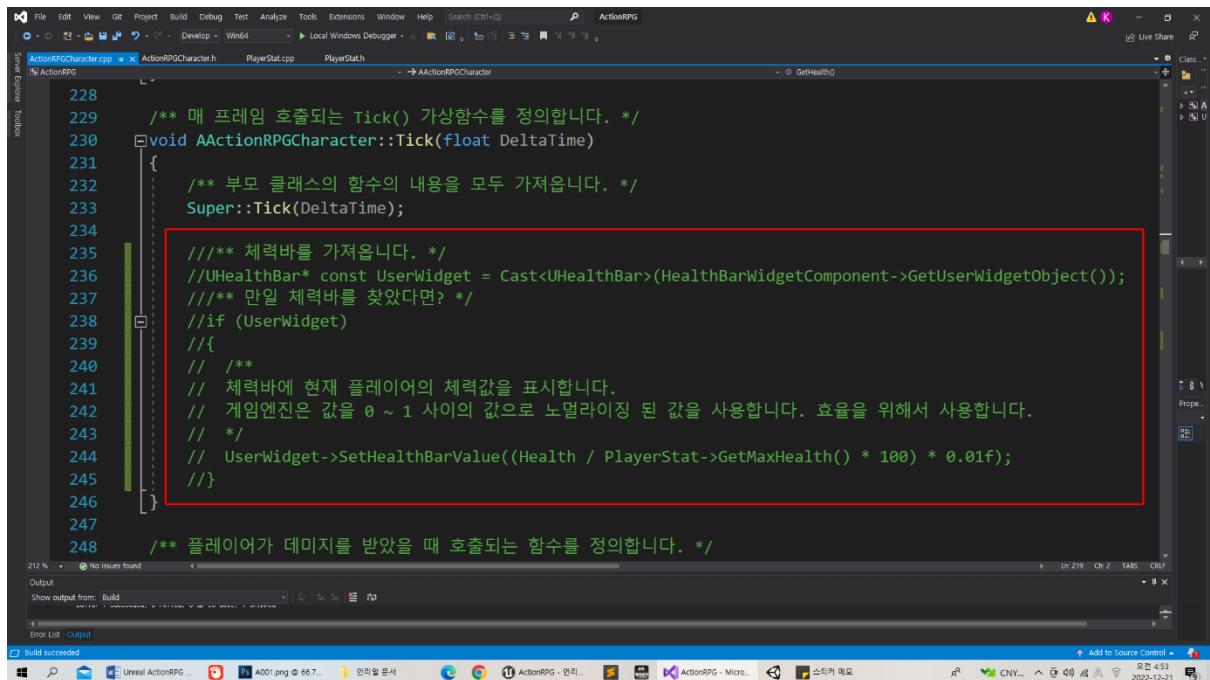


```
347
348     /** 부모클래스의 BeginPlay() 함수를 정의합니다. */
349     void AActionRPGCharacter::BeginPlay()
350     {
351         /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
352         Super::BeginPlay();
353
354         /** 플레이어의 무기들을 생성하고 기본 무기를 플레이어에게 장착합니다. */
355         SpawnPlayerWeapon();
356
357         /**
358         //캐릭터의 체력값을 테스트로 500으로 설정해 줍니다.
359         //코드상에 수치값을 직접 넣는 것을 하드코딩이라고 합니다. 지뢰밭입니다.
360         //이후에는 엑셀테이블이나 ActorComponent를 사용해서 하드코딩 안할 예정입니다.
361         /**
362         //Health = 500.0f;
363         /** 지금은 하드코딩 하고 있는데, ActorComponent를 사용해서 변수로 모두 뺄 예정입니다. */
364         /** 플레이어의 최대 체력값*/
365         // MaxHealth = 13000.0f;
366         // Health = MaxHealth;
367         Health = PlayerStat->GetMaxHealth();
368
369     }
```

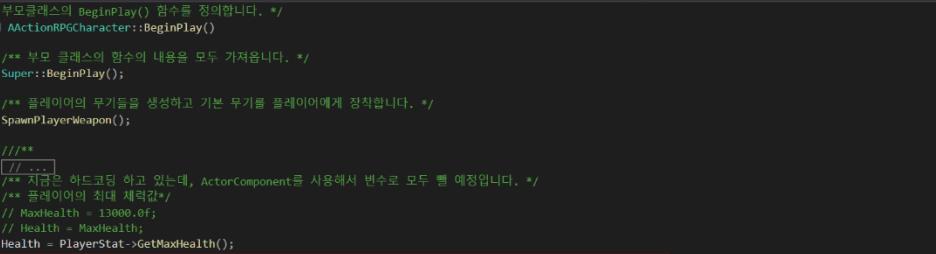


```
228  /** 매 프레임 호출되는 Tick() 가상함수를 정의합니다. */
229  void AActionRPGCharacter::Tick(float DeltaTime)
230  {
231      /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
232      Super::Tick(DeltaTime);
233
234      /** 체력바를 가져옵니다. */
235      UHealthBar* const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent->GetUserWidgetObject());
236      /** 만일 체력바를 찾았다면. */
237      if (UserWidget)
238      {
239          /**
240          체력바에 현재 플레이어의 체력값을 표시합니다.
241          게임엔진은 값을 0 ~ 1 사이의 값으로 노멀라이징 된 값을 사용합니다. 효율을 위해서 사용합니다.
242          */
243          UserWidget->SetHealthBarValue((Health / PlayerStat->GetMaxHealth() * 100) * 0.01f);
244      }
245  }
246
247
248  /** 플레이어가 데미지를 받았을 때 호출되는 함수를 정의합니다. */
249  void AActionRPGCharacter::Hit()
250  {
251  }
```

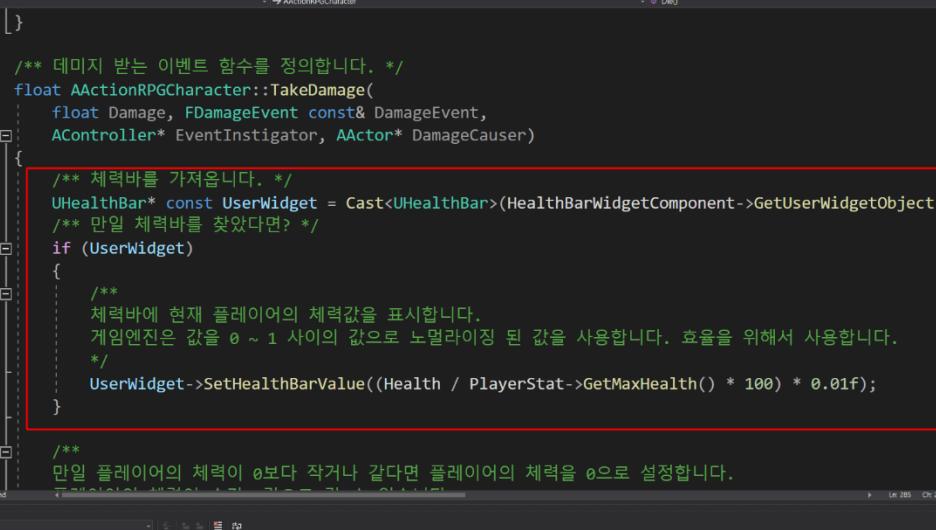
Tick() 함수 안에서 처리하면 부하가 많이 걸립니다. TakeDamage 함수안에서 처리해 주도록 합니다.



```
228  /** 매 프레임 호출되는 Tick() 가상함수를 정의합니다. */
229  void AActionRPGCharacter::Tick(float DeltaTime)
230  {
231      /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
232      Super::Tick(DeltaTime);
233
234      /**
235      // 체력바를 가져옵니다.
236      //UHealthBar* const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent->GetUserWidgetObject());
237      ///** 만일 체력바를 찾았다면. */
238      //if (UserWidget)
239      //{
240          /**
241          // 체력바에 현재 플레이어의 체력값을 표시합니다.
242          // 게임엔진은 값을 0 ~ 1 사이의 값으로 노멀라이징 된 값을 사용합니다. 효율을 위해서 사용합니다.
243          // */
244          // UserWidget->SetHealthBarValue((Health / PlayerStat->GetMaxHealth() * 100) * 0.01f);
245      //}
246  }
247
248  /** 플레이어가 데미지를 받았을 때 호출되는 함수를 정의합니다. */
249
```

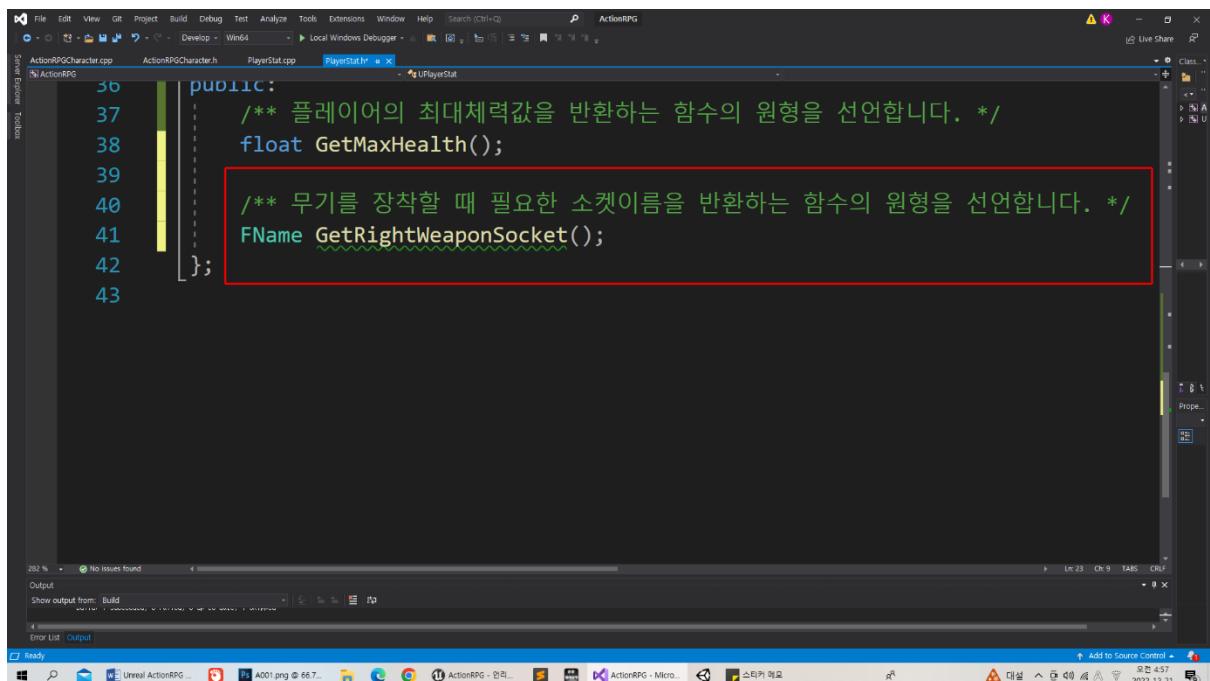


```
360     /** 부모클래스의 BeginPlay() 함수를 정의합니다. */
361     void AACTIONRPGCharacter::BeginPlay()
362     {
363         /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
364         Super::BeginPlay();
365
366         /** 플레이어의 무기들을 생성하고 기본 무기를 플레이어에게 장착합니다. */
367         SpawnPlayerWeapon();
368
369         //////
370         // ...
371         /** 지금은 하드코딩 하고 있는데, ActorComponent를 사용해서 변수로 모두 뺄 예정입니다. */
372         /** 플레이어의 최대 체력값*/
373         // MaxHealth = 13000.0f;
374         // Health = MaxHealth;
375         Health = PlayerStat->GetMaxHealth();
376
377         /** 체력바를 가져옵니다. */
378         UHealthBar* const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent-> GetUserWidgetObject());
379         /** 만일 체력바를 찾았다면? */
380         if (UserWidget)
381         {
382             /**
383             체력바에 현재 플레이어의 체력값을 표시합니다.
384             게임엔진은 값은 0 ~ 1 사이의 값으로 노말라이징 된 값을 사용합니다. 효율을 위해서 사용합니다.
385             */
386             UserWidget->SetHealthBarValue((Health / PlayerStat->GetMaxHealth() * 100) * 0.01f);
387         }
388
389         /** 사용자가 지정한 무기를 장착하는 함수를 정의합니다. */
390     }
391
392     // ...
393
394     /** 사용자가 지정한 무기를 장착하는 함수를 정의합니다. */
395 }
```

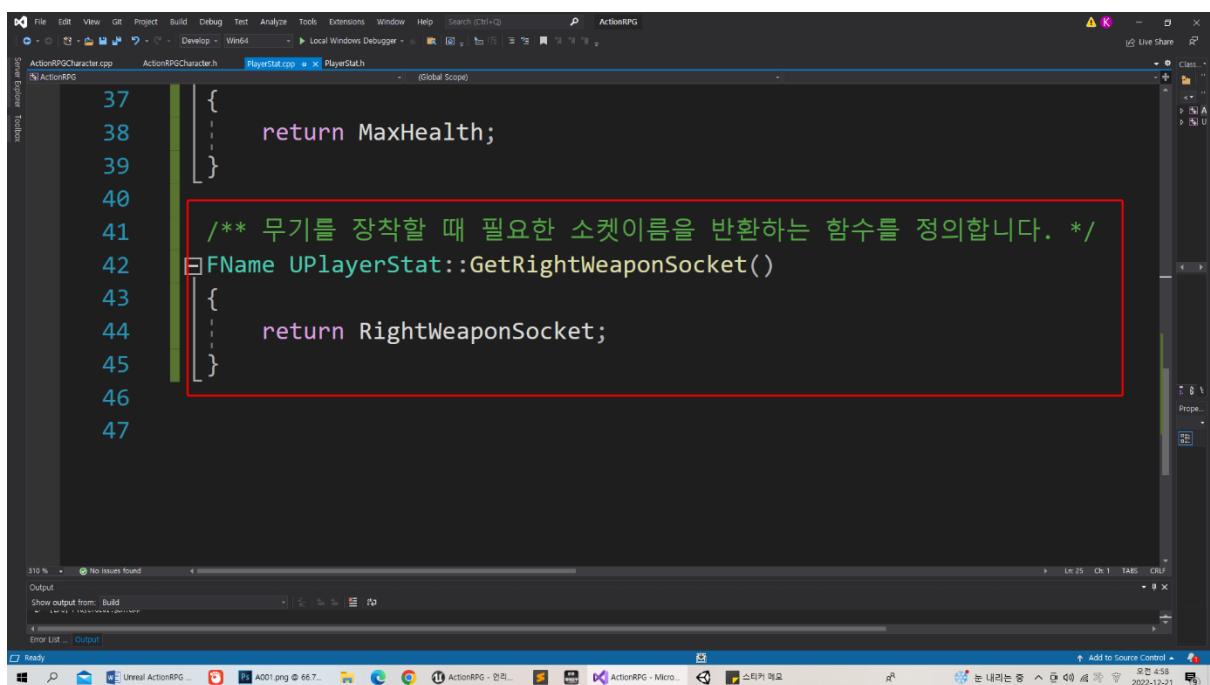


```
296     }
297
298     /** 데미지 받는 이벤트 함수를 정의합니다. */
299     float ActionRPGCharacter::TakeDamage(
300         float Damage, FDamageEvent const& DamageEvent,
301         AController* EventInstigator, AActor* DamageCauser)
302     {
303         /** 체력바를 가져옵니다. */
304         UHealthBar* const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent-> GetUserWidgetObject());
305         /** 만일 체력바를 찾았다면? */
306         if (UserWidget)
307         {
308             /**
309             체력바에 현재 플레이어의 체력값을 표시합니다.
310             게임엔진은 값을 0 ~ 1 사이의 값으로 노멀라이징 된 값을 사용합니다. 효율을 위해서 사용합니다.
311             */
312             UserWidget->SetHealthBarValue((Health / PlayerStat->GetMaxHealth() * 100) * 0.01f);
313         }
314
315         /**
316         만일 플레이어의 체력이 0보다 작거나 같다면 플레이어의 체력을 0으로 설정합니다.
317         */
318     }
319 }
```

소켓을 반환하는 함수의 원형을 선언합니다.

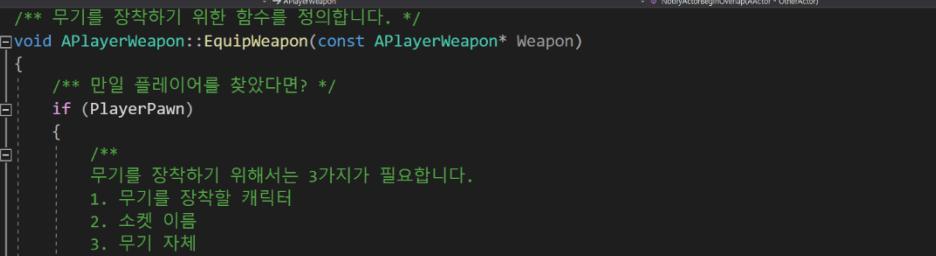


```
36     public:
37     /** 플레이어의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
38     float GetMaxHealth();
39
40     /** 무기를 장착할 때 필요한 소켓이름을 반환하는 함수의 원형을 선언합니다. */
41     FName GetRightWeaponSocket();
42
43 }
```



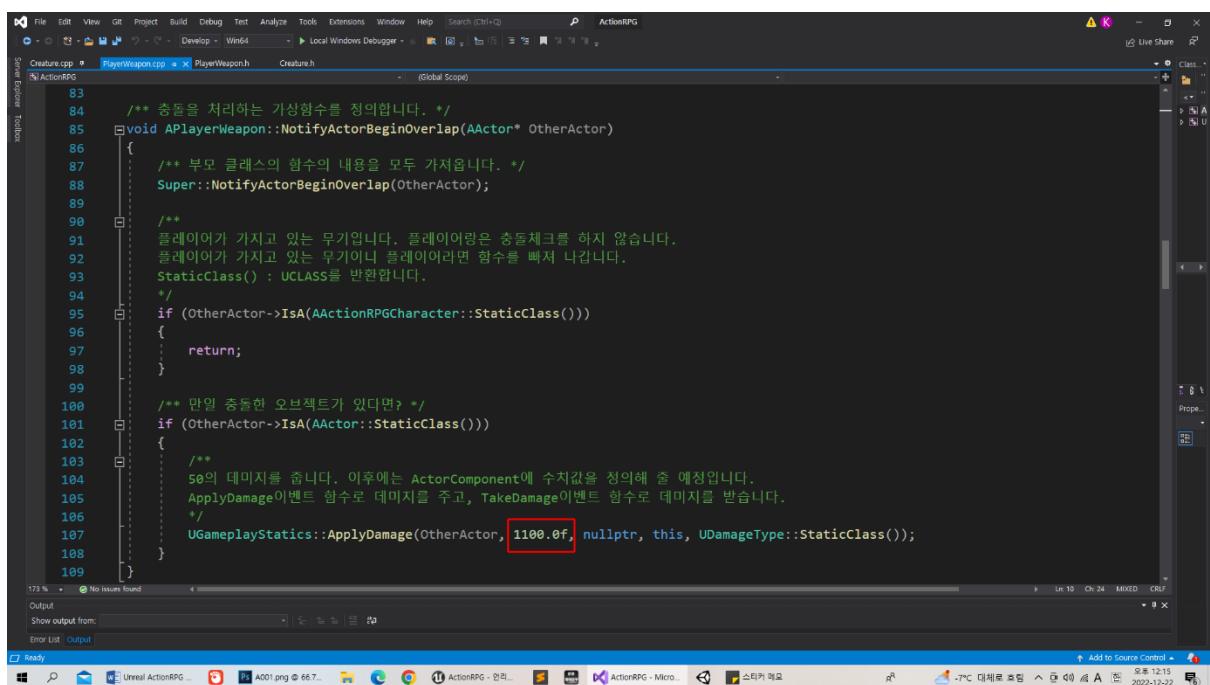
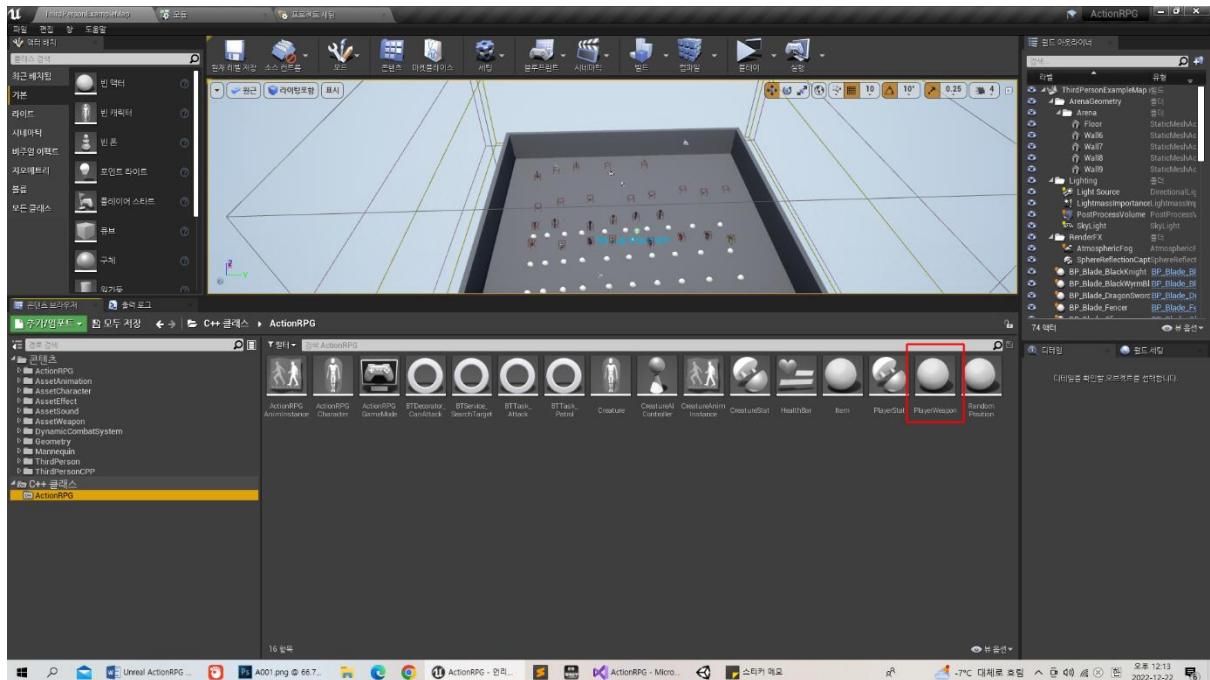
```
37     {
38     return MaxHealth;
39     }
40
41     /** 무기를 장착할 때 필요한 소켓이름을 반환하는 함수를 정의합니다. */
42     FName UPlayerStat::GetRightWeaponSocket()
43     {
44     return RightWeaponSocket;
45     }
46
47 }
```

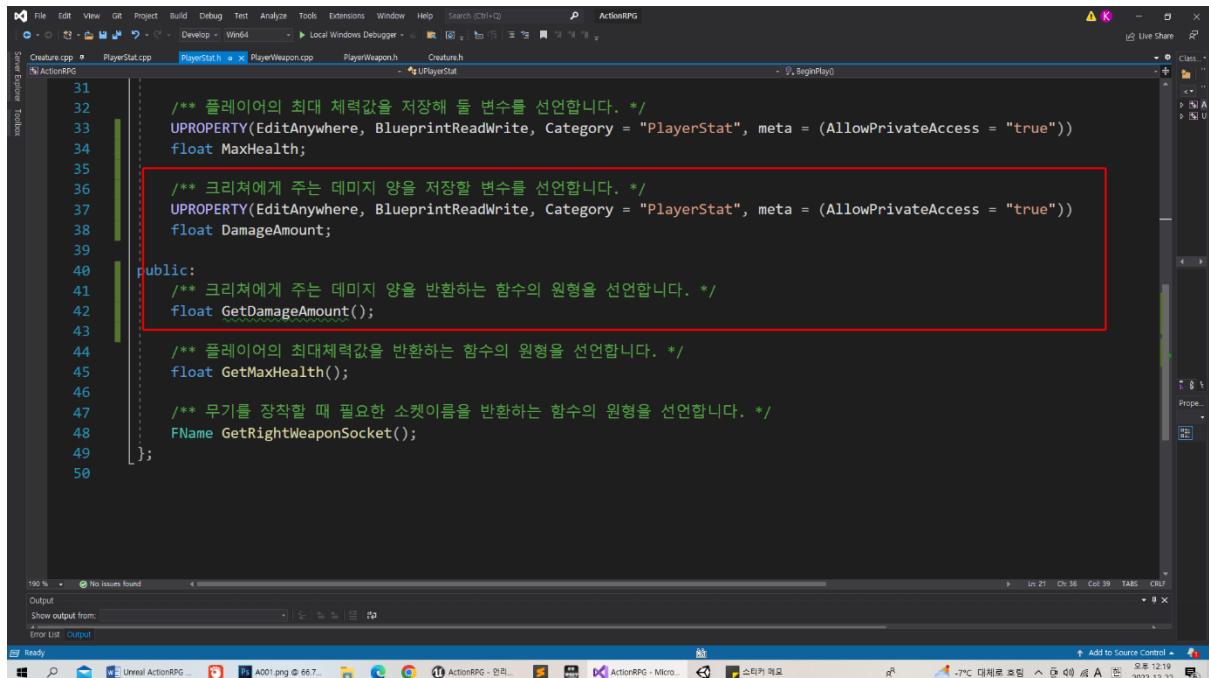
수정해 주도록 합니다.



```
117     /** 무기를 장착하기 위한 합수를 정의합니다. */
118     void APlayerWeapon::EquipWeapon(const APlayerWeapon* Weapon)
119     {
120         /** 만일 플레이어를 찾았다면? */
121         if (PlayerPawn)
122         {
123             /**
124             무기를 장착하기 위해서는 3가지가 필요합니다.
125             1. 무기를 장착할 캐릭터
126             2. 소켓 이름
127             3. 무기 자체
128             */
129             /**
130             플레이어의 스케레탈메쉬를 가져옵니다.
131             USkeletalMeshComponent* PlayerMesh = PlayerPawn->GetMesh();
132             /**
133             플레이어의 소켓이름을 가져옵니다.
134             FName WeaponSocket = PlayerPawn->PlayerStat->GetRightWeaponSocket();
135             /**
136             무기를 플레이어에게 볼어 줍니다.
137             WeaponMesh->AttachTo(PlayerMesh, WeaponSocket);
138         }
139     }
```

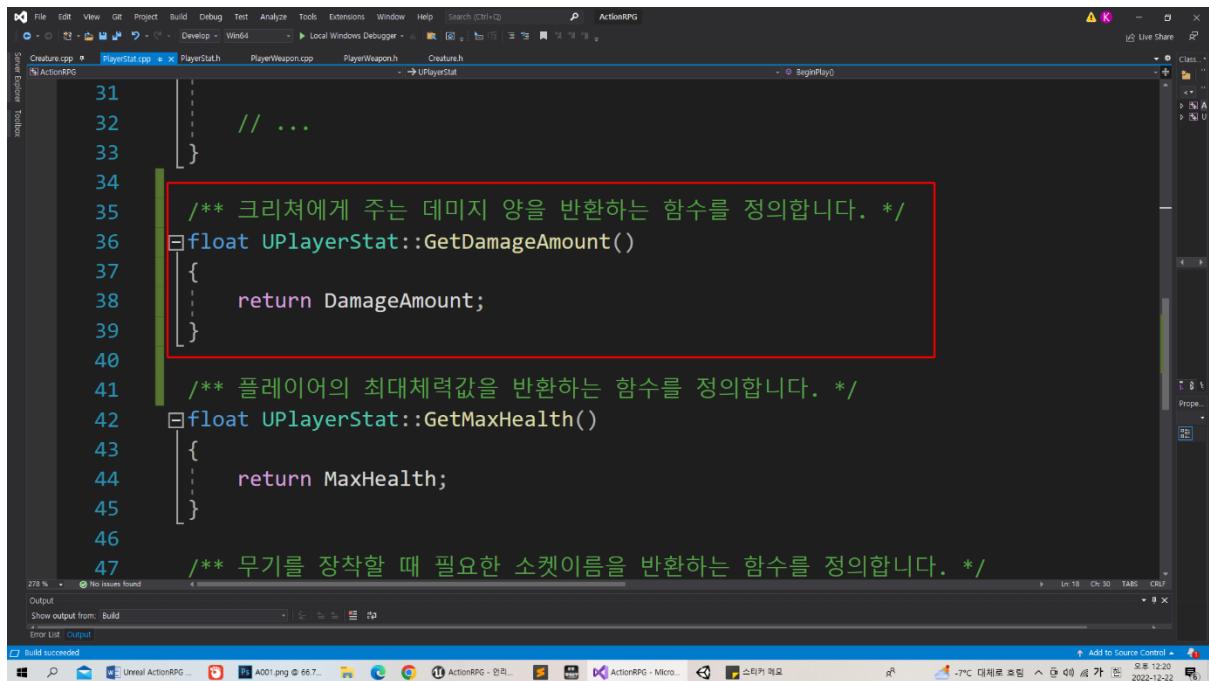
크리쳐에게 데미지를 주는 값을 적용해 주도록 합니다.





```
31  /** 플레이어의 최대 체력값을 저장해 둘 변수를 선언합니다. */
32  UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "PlayerStat", meta = (AllowPrivateAccess = "true"))
33  float MaxHealth;
34
35  /** 크리쳐에게 주는 데미지 양을 저장할 변수를 선언합니다. */
36  UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "PlayerStat", meta = (AllowPrivateAccess = "true"))
37  float DamageAmount;
38
39  public:
40  /** 크리쳐에게 주는 데미지 양을 반환하는 함수의 원형을 선언합니다. */
41  float GetDamageAmount();
42
43  /** 플레이어의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
44  float GetMaxHealth();
45
46  /** 무기를 장착할 때 필요한 소켓이름을 반환하는 함수의 원형을 선언합니다. */
47  FName GetRightWeaponSocket();
48
49 }
50
```

정의해 줍니다.



```
31
32  // ...
33 }
34
35  /** 크리쳐에게 주는 데미지 양을 반환하는 함수를 정의합니다. */
36  float UPlayerStat::GetDamageAmount()
37  {
38  return DamageAmount;
39  }
40
41  /** 플레이어의 최대체력값을 반환하는 함수를 정의합니다. */
42  float UPlayerStat::GetMaxHealth()
43  {
44  return MaxHealth;
45  }
46
47  /** 무기를 장착할 때 필요한 소켓이름을 반환하는 함수를 정의합니다. */
48
```

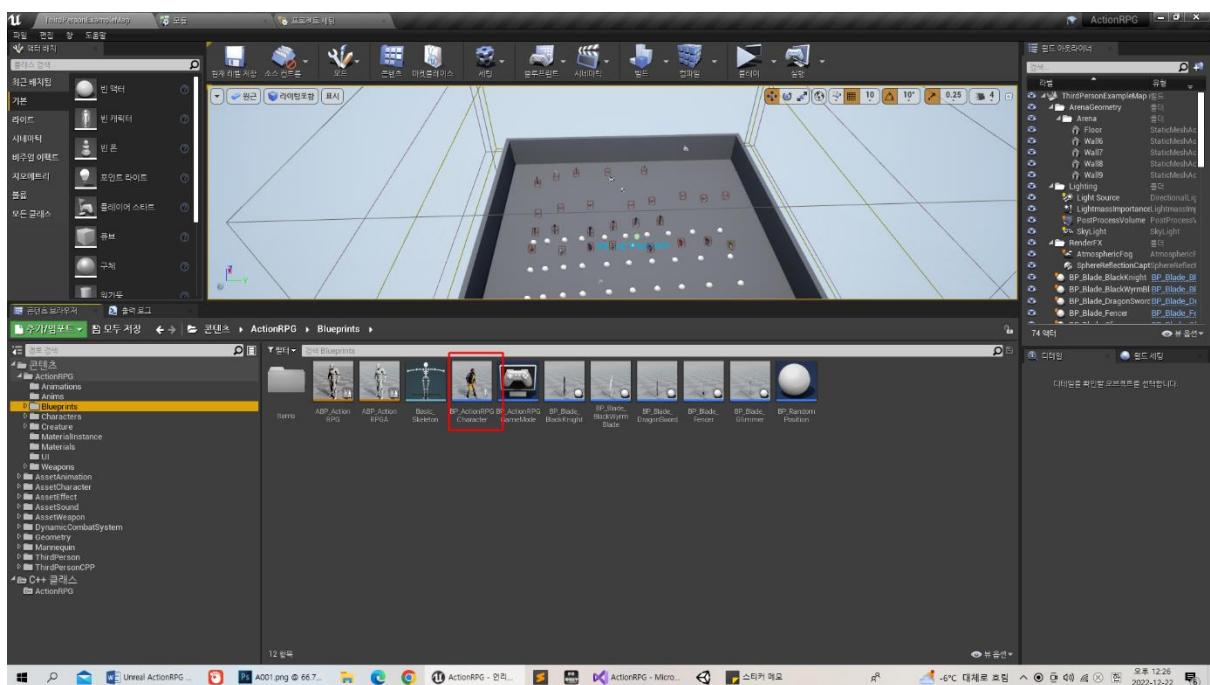
PlayerWeapon 클래스에서 수정해 주도록 합니다.

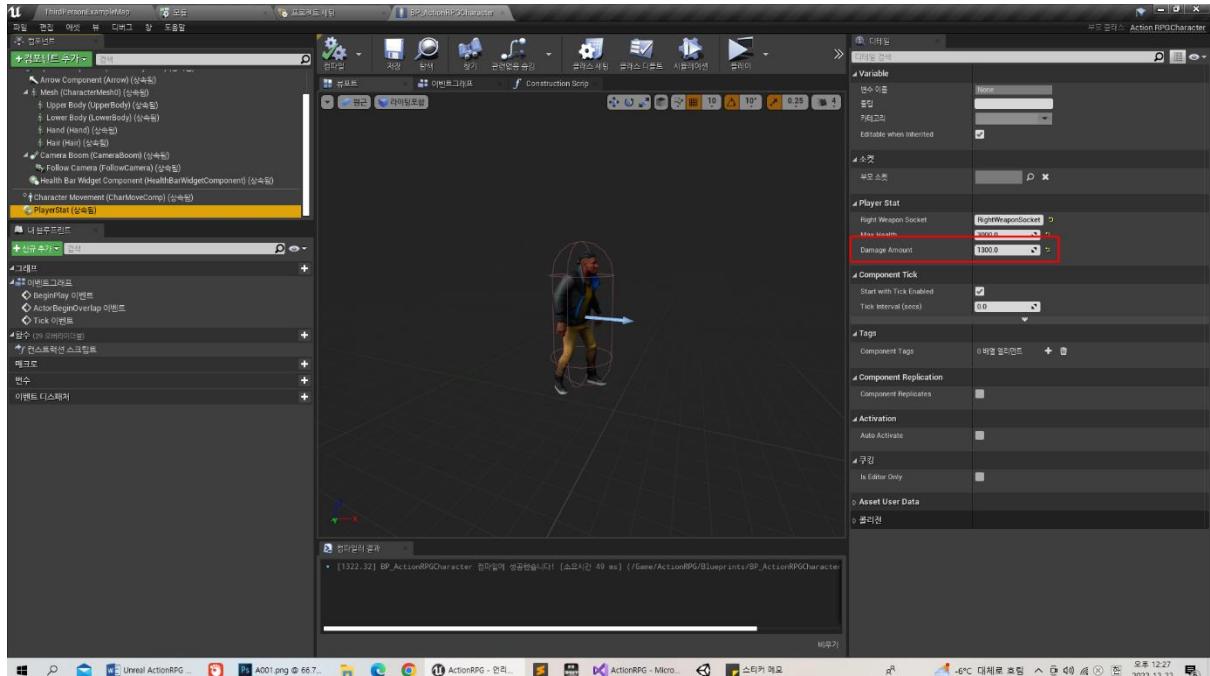
Screenshot of the Unreal Engine 4 Editor showing the C++ code for PlayerWeapon.h. The code handles collision detection and damage application. A red box highlights the section where damage is applied to other actors.

```

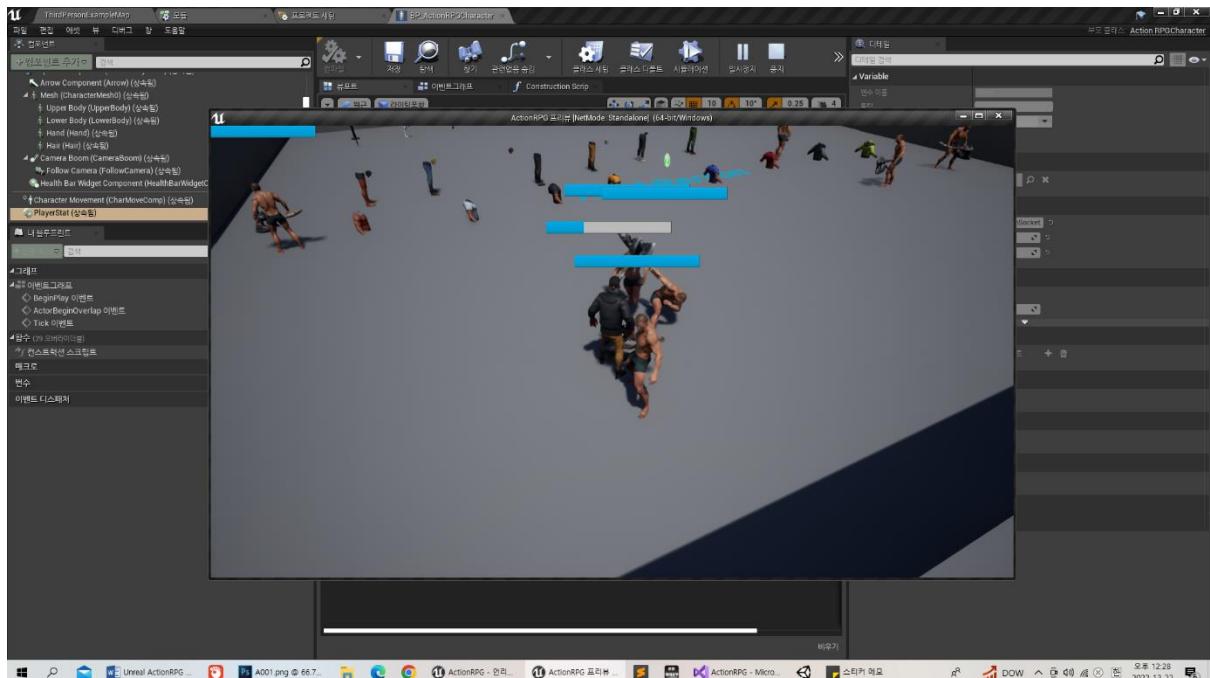
83  /** 충돌을 처리하는 기상함수를 정의합니다. */
84  void APlayerWeapon::NotifyActorBeginOverlap(AActor* OtherActor)
85  {
86      /* 부모 클래스의 함수의 내용을 모두 가져옵니다. */
87      Super::NotifyActorBeginOverlap(OtherActor);
88
89      /**
90      플레이어가 가지고 있는 무기입니다. 플레이어는 충돌체크를 하지 않습니다.
91      플레이어가 가지고 있는 무기이니 플레이어라면 함수를 빼서 나갑니다.
92      StaticClass() : UCLASS를 반환합니다.
93      */
94      if (OtherActor->IsA(AActionRPGCharacter::StaticClass()))
95      {
96          return;
97      }
98
99      /**
100      만약 충돌한 오브젝트가 있다면 */
101     if (OtherActor->IsA(AActor::StaticClass()))
102     {
103         /**
104         so의 데미지를 줍니다. 이후에는 ActorComponent에 수치값을 정의해 줄 예정입니다.
105         ApplyDamage이벤트 함수로 데미지를 주고, TakeDamage이벤트 함수로 데미지를 받습니다.
106         */
107         /**
108         플레이어를 찾습니다. */
109         AActionRPGCharacter* PlayerAvatar = Cast<AActionRPGCharacter>(UGameplayStatics::GetPlayerCharacter(GetWorld(), 0));
110         /** 데미지 양을 시정해 줍니다. */
111         float DamageAmount = PlayerAvatar->PlayerStat->GetDamageAmount();
112         UGameplayStatics::ApplyDamage(OtherActor, DamageAmount, nullptr, this, UDamageType::StaticClass());
113     }
114
115     /** bIsAttack 블리언 변수를 반환하는 함수를 정의합니다. */
116 }

```

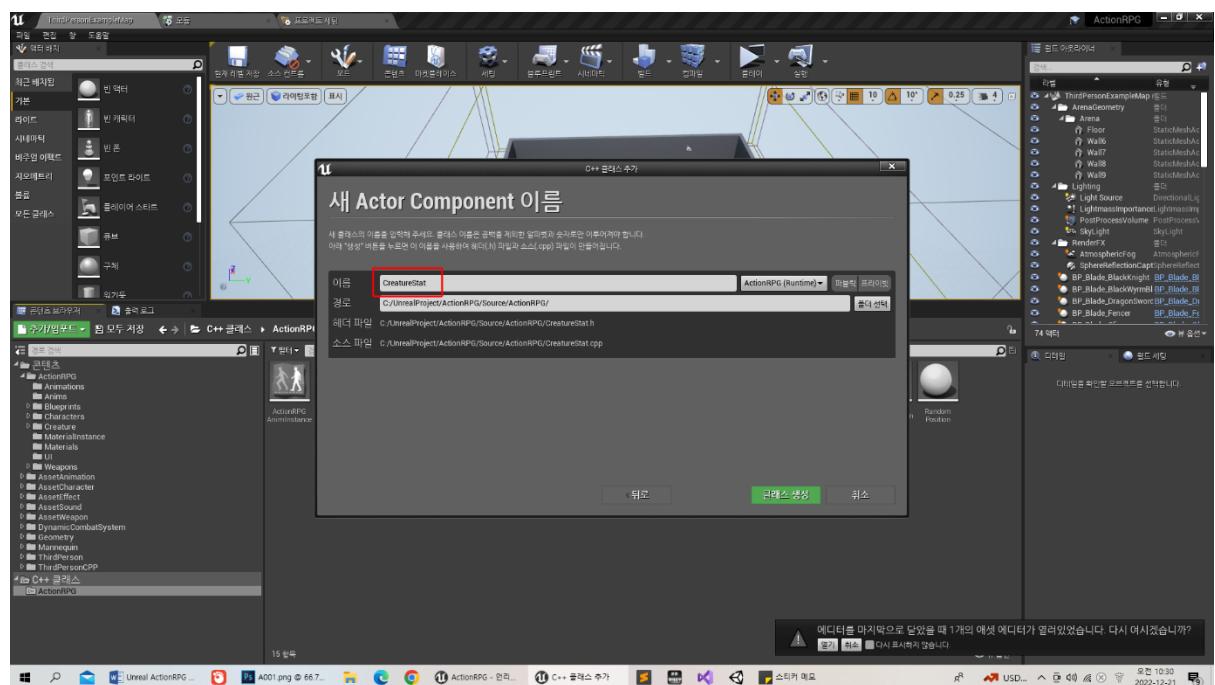
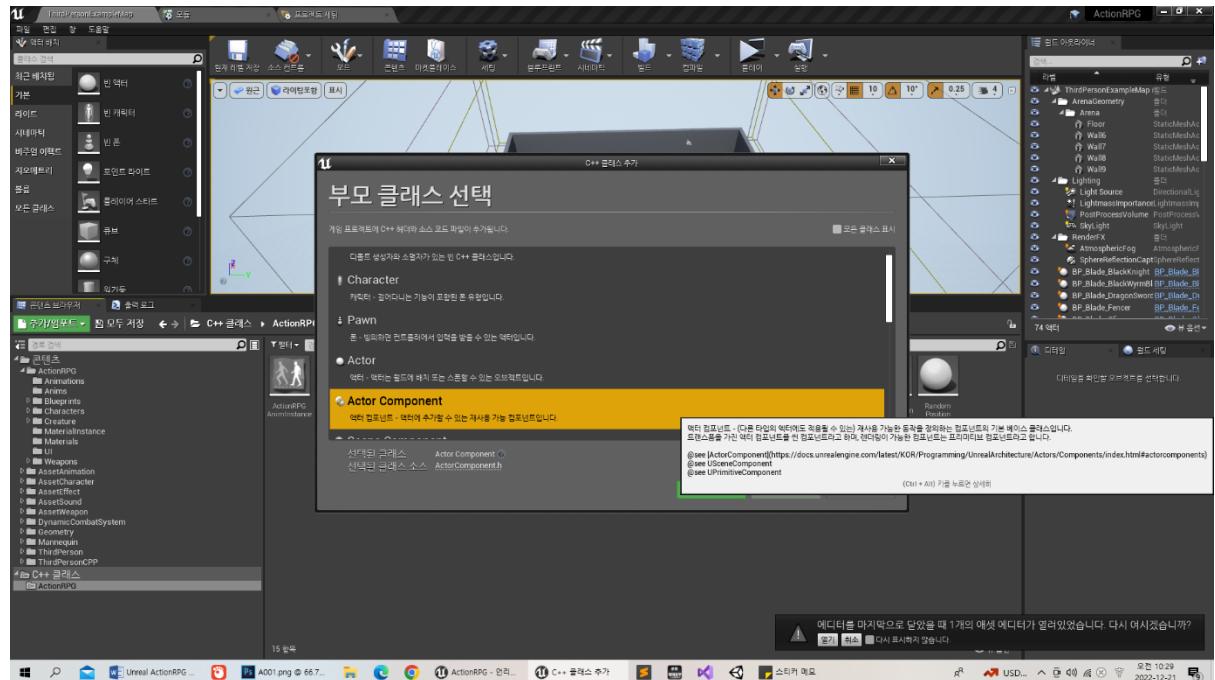


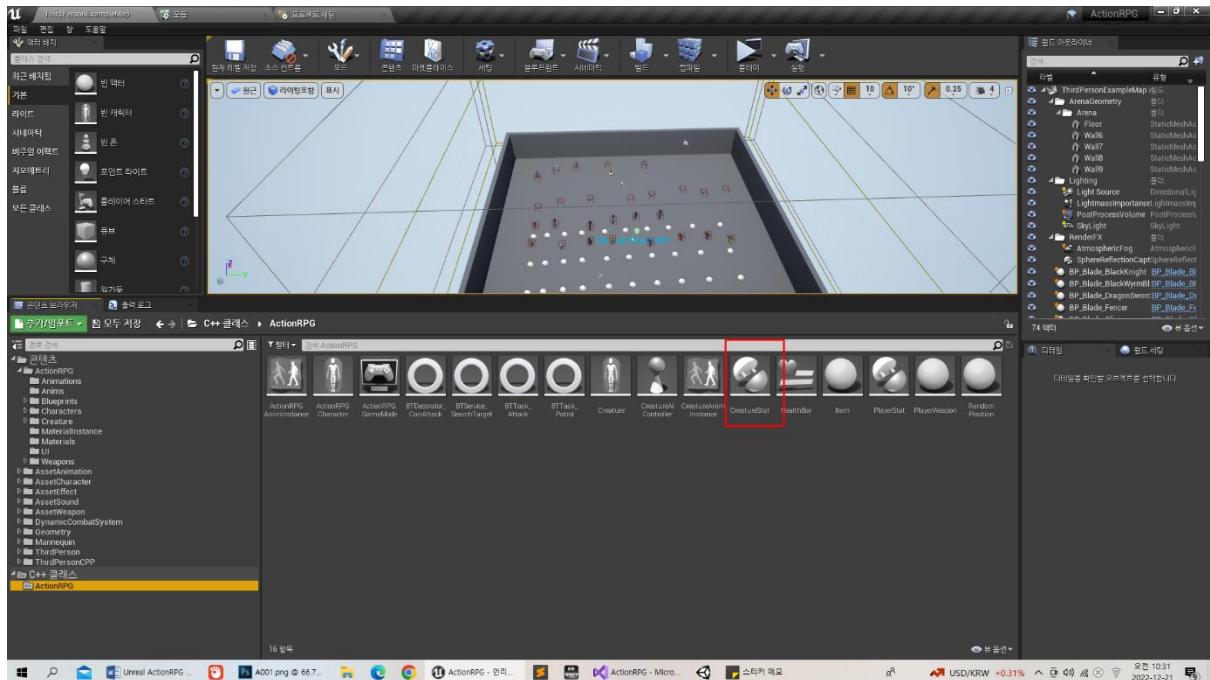


플레이를 해서 결과를 확인합니다.

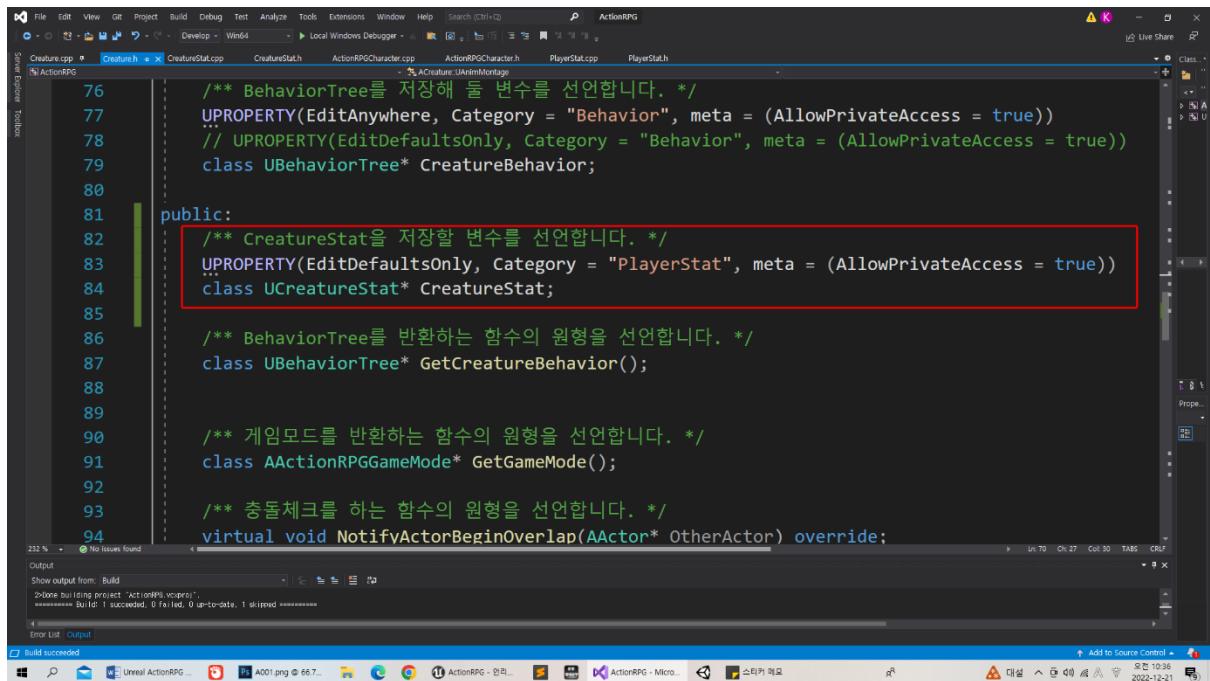


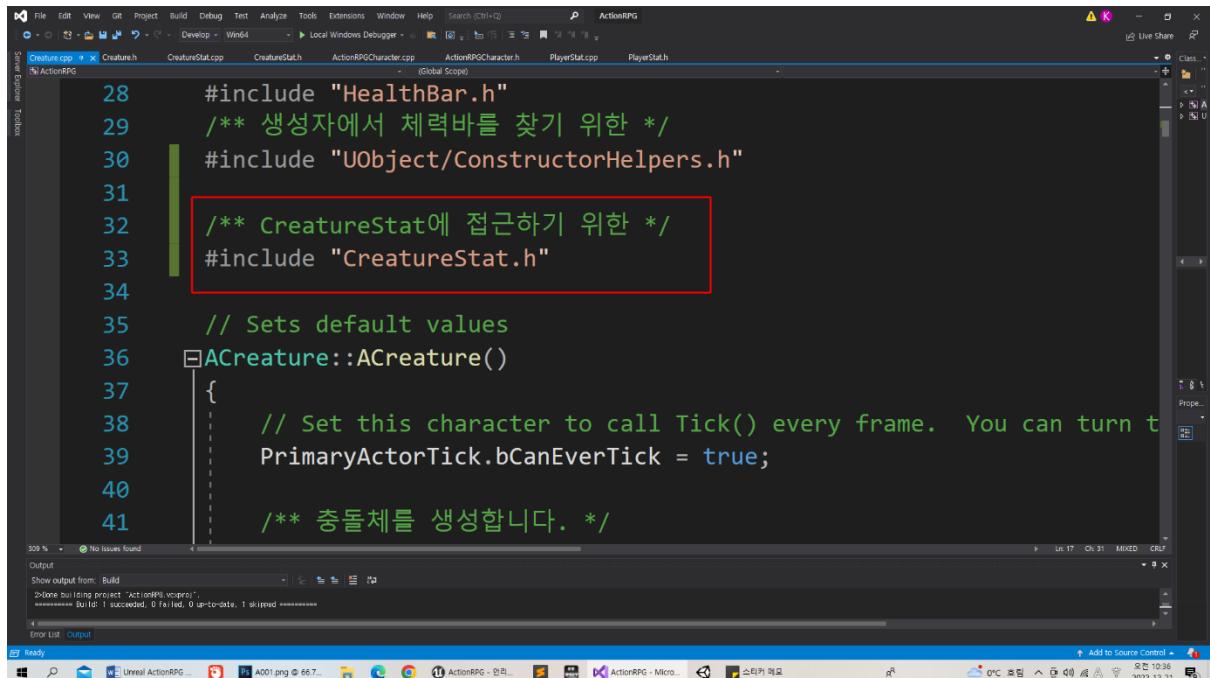
이제 크리쳐쪽도 해 주도록 합니다. ,ActorComponent를 부모 클래스로 선택하는 CreatureStat이라 는 클래스를 정의해 주도록 합니다.



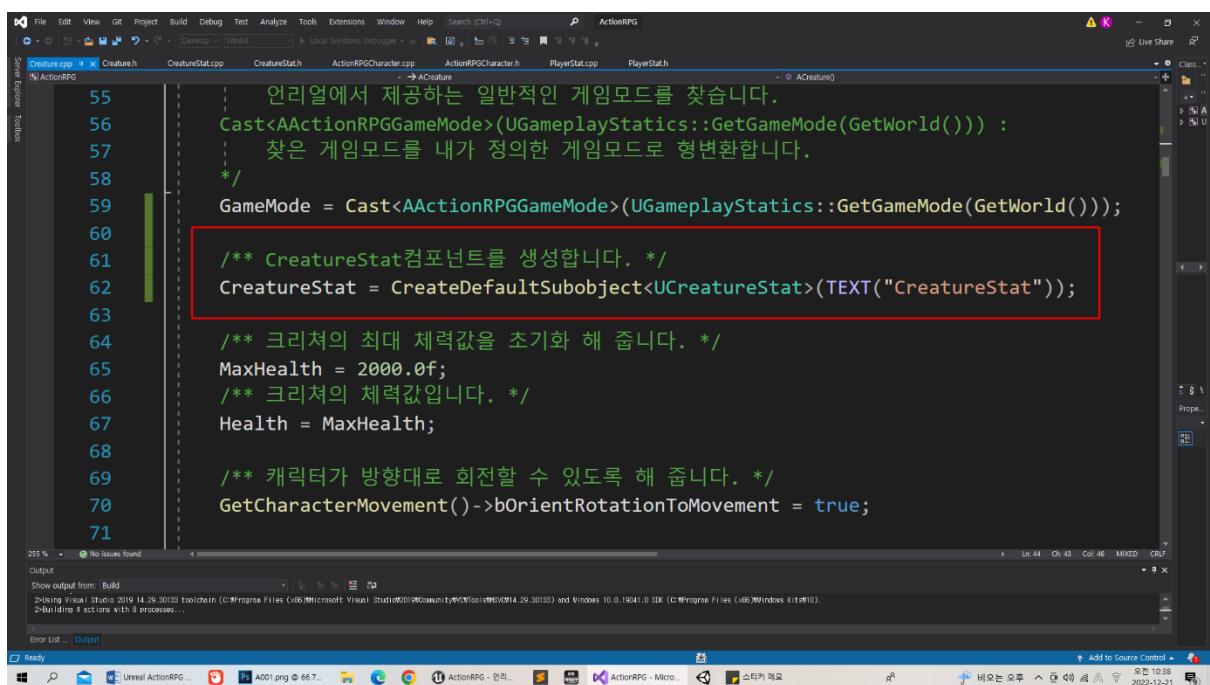


생성자 함수에서 생성해 주도록 합니다.



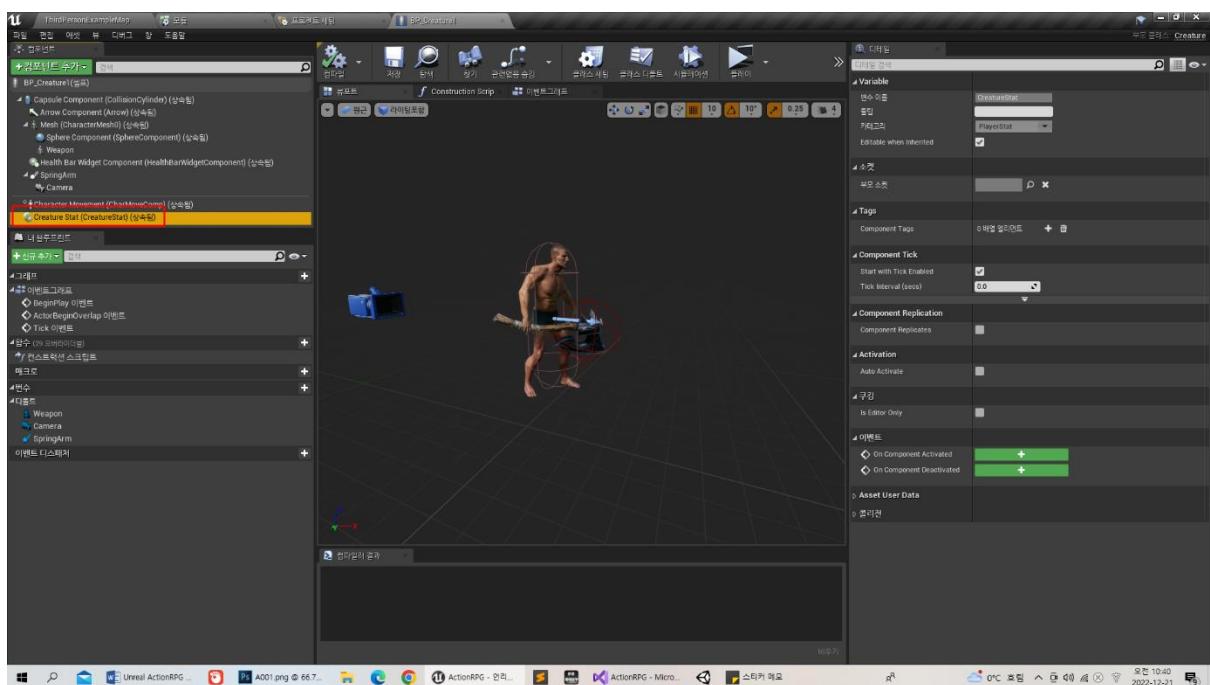
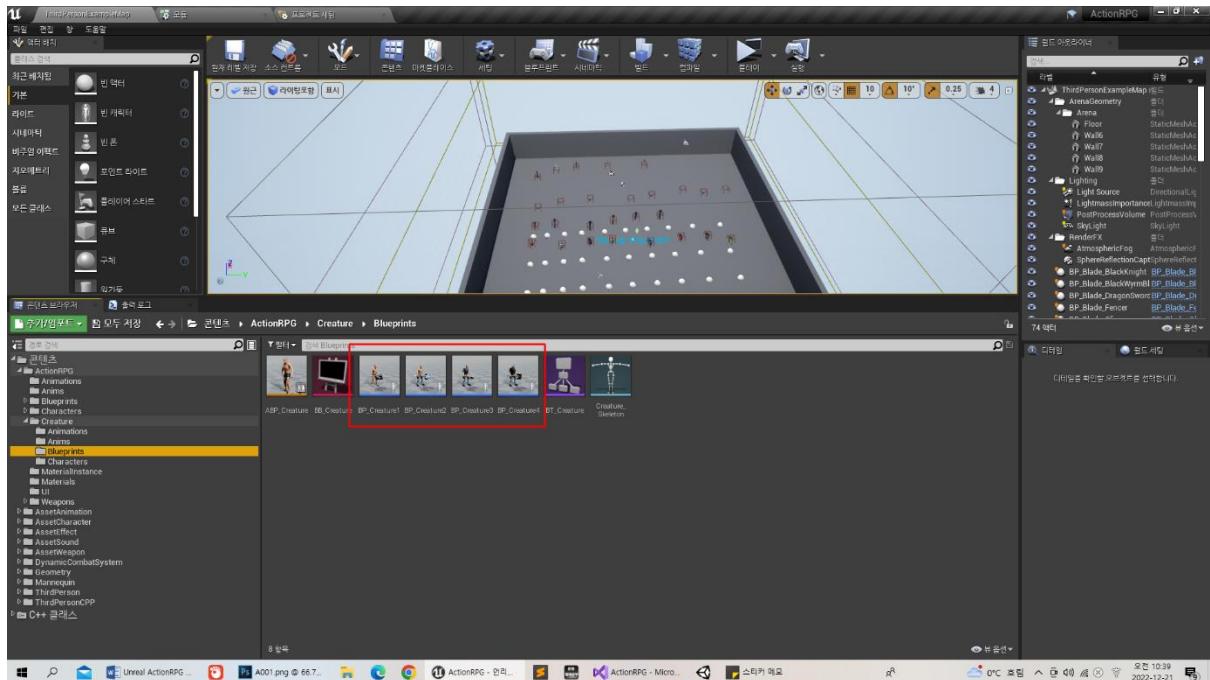


```
28     #include "HealthBar.h"
29     /** 생성자에서 체력바를 찾기 위한 */
30     #include "UObject/ConstructorHelpers.h"
31
32     /** CreatureStat에 접근하기 위한 */
33     #include "CreatureStat.h"
34
35     // Sets default values
36     ACreature::ACreature()
37     {
38         // Set this character to call Tick() every frame. You can turn this off in the Settings panel
39         PrimaryActorTick.bCanEverTick = true;
40
41         /** 충돌체를 생성합니다. */
```

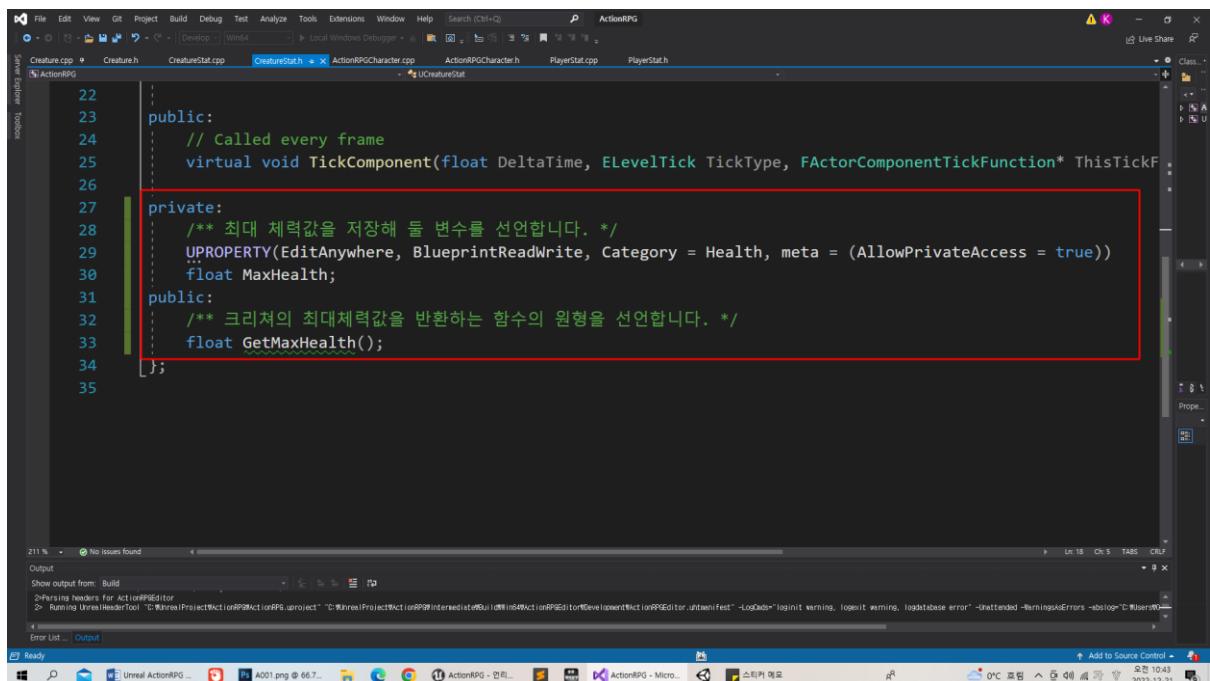


```
55     언리얼에서 제공하는 일반적인 게임모드를 찾습니다.
56     Cast<AACTIONRGGameMode>(UGameplayStatics::GetGameMode(GetWorld())) :
57         찾은 게임모드를 내가 정의한 게임모드로 형변환합니다.
58     */
59     GameMode = Cast<AACTIONRGGameMode>(UGameplayStatics::GetGameMode(GetWorld()));
60
61     /** CreatureStat컴포넌트를 생성합니다. */
62     CreatureStat = CreateDefaultSubobject<UCreatureStat>(TEXT("CreatureStat"));
63
64     /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
65     MaxHealth = 2000.0f;
66     /** 크리쳐의 체력값입니다. */
67     Health = MaxHealth;
68
69     /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
70     GetCharacterMovement()->bOrientRotationToMovement = true;
71
```

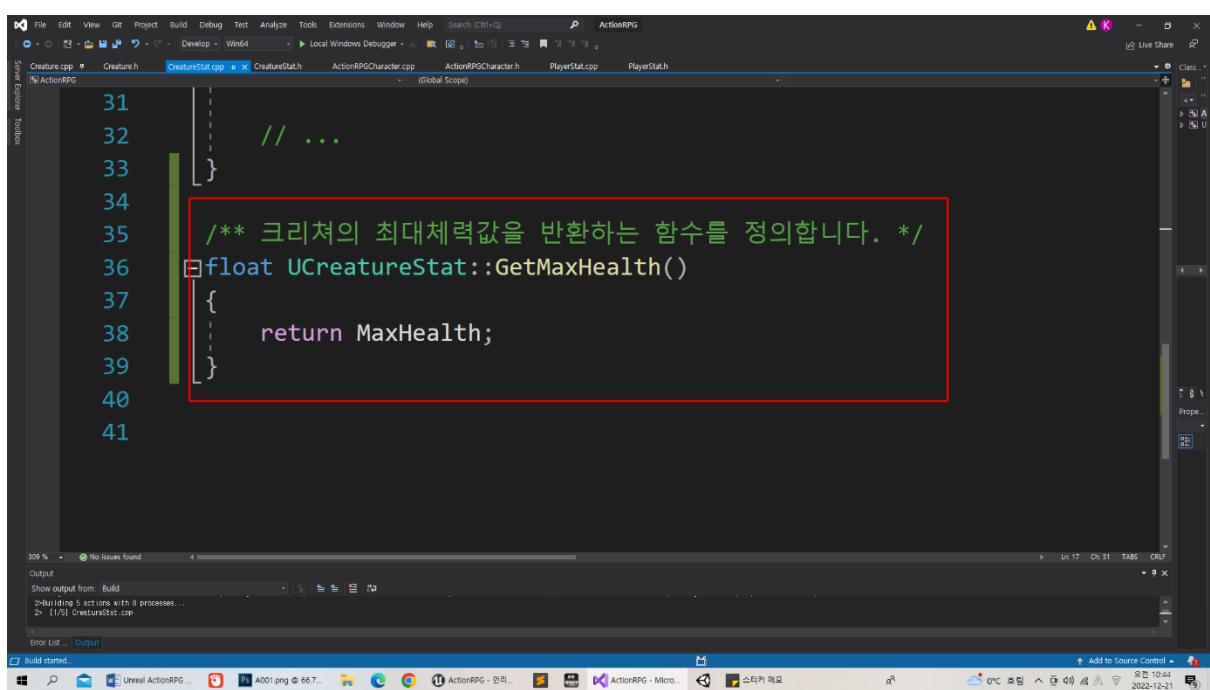
결과를 확인합니다.



크리쳐의 체력부터 적용해 주도록 합니다.

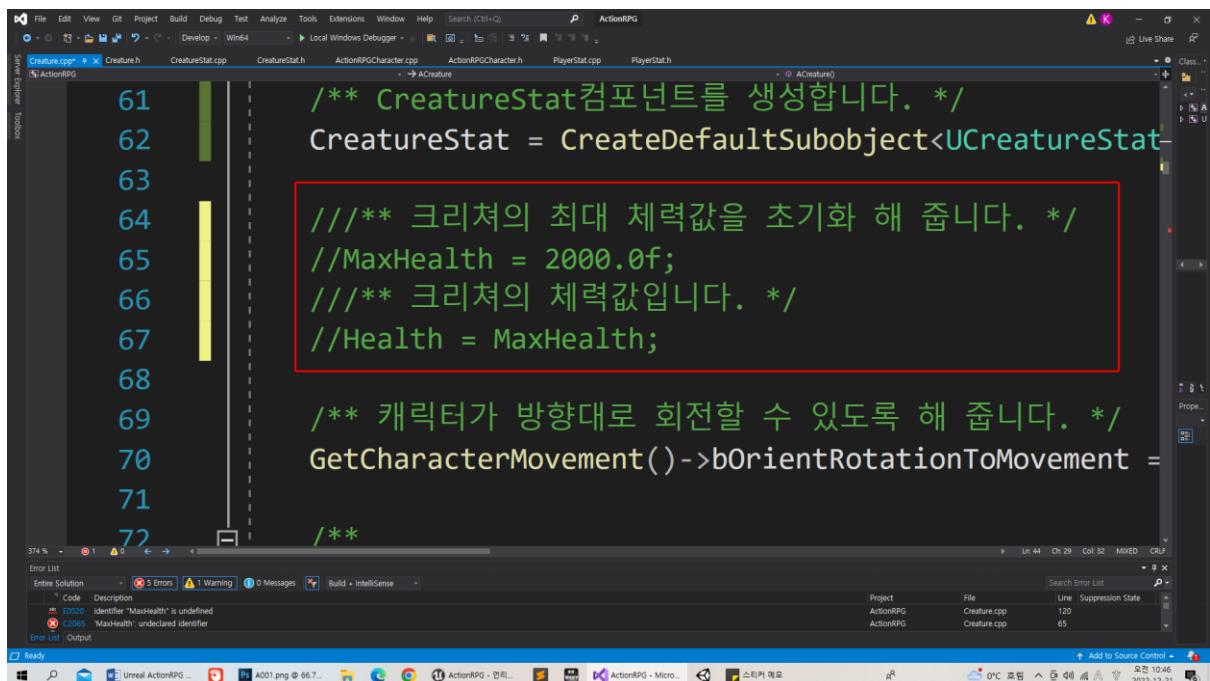


```
22
23     public:
24         // Called every frame
25         virtual void TickComponent(float DeltaTime, ELevelTick TickType, FActorComponentTickFunction* ThisTickFunction) override;
26
27     private:
28         /** 최대 체력값을 저장해 둘 변수를 선언합니다. */
29         UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = Health, meta = (AllowPrivateAccess = true))
30         float MaxHealth;
31
32     public:
33         /** 크리쳐의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
34         float GetMaxHealth();
35     
```



```
31
32     // ...
33
34
35     /** 크리쳐의 최대체력값을 반환하는 함수를 정의합니다. */
36     float UCreatureStat::GetMaxHealth()
37     {
38         return MaxHealth;
39     }
40
41

```



```
61  /** CreatureStat컴포넌트를 생성합니다. */
62  CreatureStat = CreateDefaultSubobject<UCreatureStat>(CreatureStatName);
63
64  // /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
65  // MaxHealth = 2000.0f;
66  // /** 크리쳐의 체력값입니다. */
67  // Health = MaxHealth;
68
69  /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
70  GetCharacterMovement()->bOrientRotationToMovement = true;
71
72  /**
73
74  */

75  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
76  // GetCharacterMovement()->bOrientRotationToMovement = true;
77
78  /**
79
80  */

81  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
82  // GetCharacterMovement()->bOrientRotationToMovement = true;
83
84  /**
85
86  */

87  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
88  // GetCharacterMovement()->bOrientRotationToMovement = true;
89
90  /**
91
92  */

93  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
94  // GetCharacterMovement()->bOrientRotationToMovement = true;
95
96  /**
97
98  */

99  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
100 // GetCharacterMovement()->bOrientRotationToMovement = true;
101
102 // Called when the game starts or when spawned
103 void ACreature::BeginPlay()
104 {
105     Super::BeginPlay();
106
107     // /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
108     // MaxHealth = 2000.0f;
109     // /** 크리쳐의 체력값입니다. */
110     // Health = MaxHealth;
111     Health = CreatureStat->GetMaxHealth();
112 }
```

Screenshot of the Unreal Engine Editor showing the Creature.cpp code. A red box highlights the assignment of MaxHealth to Health. The code is as follows:

```
61  /** CreatureStat컴포넌트를 생성합니다. */
62  CreatureStat = CreateDefaultSubobject<UCreatureStat>(CreatureStatName);
63
64  // /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
65  // MaxHealth = 2000.0f;
66  // /** 크리쳐의 체력값입니다. */
67  // Health = MaxHealth;
68
69  /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
70  GetCharacterMovement()->bOrientRotationToMovement = true;
71
72  /**
73
74  */

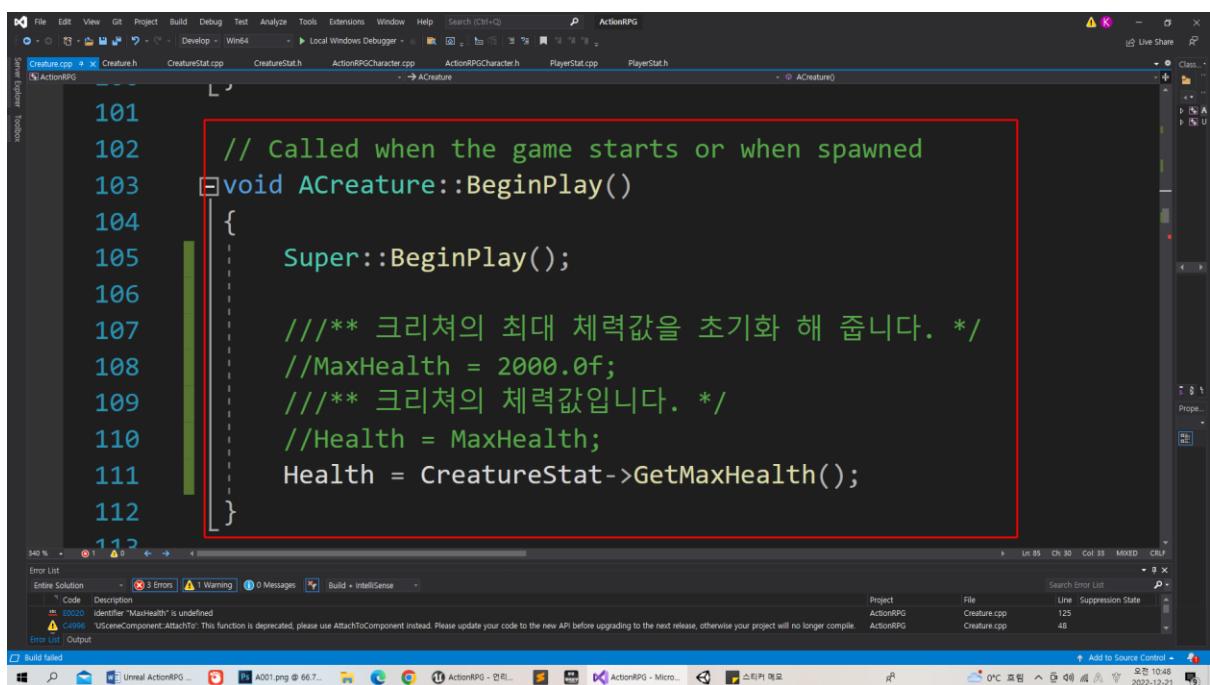
75  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
76  // GetCharacterMovement()->bOrientRotationToMovement = true;
77
78  /**
79
80  */

81  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
82  // GetCharacterMovement()->bOrientRotationToMovement = true;
83
84  /**
85
86  */

87  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
88  // GetCharacterMovement()->bOrientRotationToMovement = true;
89
90  /**
91
92  */

93  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
94  // GetCharacterMovement()->bOrientRotationToMovement = true;
95
96  /**
97
98  */

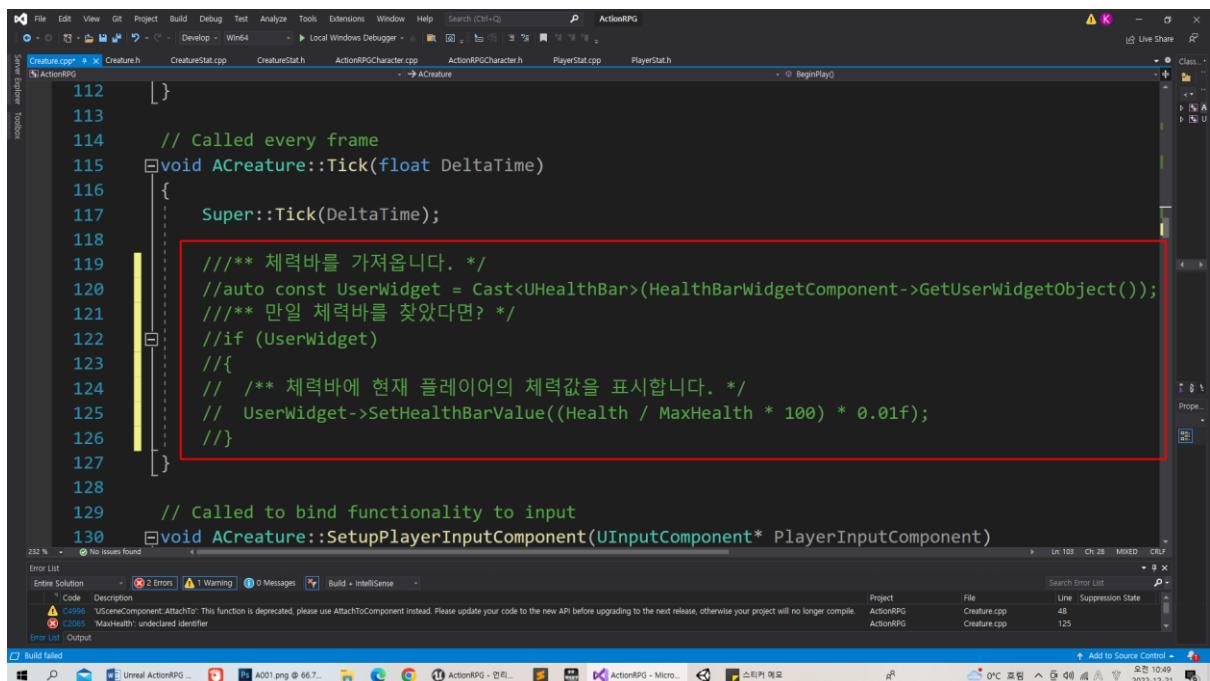
99  // /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
100 // GetCharacterMovement()->bOrientRotationToMovement = true;
101
102 // Called when the game starts or when spawned
103 void ACreature::BeginPlay()
104 {
105     Super::BeginPlay();
106
107     // /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
108     // MaxHealth = 2000.0f;
109     // /** 크리쳐의 체력값입니다. */
110     // Health = MaxHealth;
111     Health = CreatureStat->GetMaxHealth();
112 }
```



```
101
102 // Called when the game starts or when spawned
103 void ACreature::BeginPlay()
104 {
105     Super::BeginPlay();
106
107     // /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
108     // MaxHealth = 2000.0f;
109     // /** 크리쳐의 체력값입니다. */
110     // Health = MaxHealth;
111     Health = CreatureStat->GetMaxHealth();
112 }
```

Screenshot of the Unreal Engine Editor showing the Creature.cpp code. A red box highlights the call to GetMaxHealth() in the BeginPlay() function. The code is as follows:

```
101
102 // Called when the game starts or when spawned
103 void ACreature::BeginPlay()
104 {
105     Super::BeginPlay();
106
107     // /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
108     // MaxHealth = 2000.0f;
109     // /** 크리쳐의 체력값입니다. */
110     // Health = MaxHealth;
111     Health = CreatureStat->GetMaxHealth();
112 }
```



```
112     }
113
114     // Called every frame
115     void ACreature::Tick(float DeltaTime)
116     {
117         Super::Tick(DeltaTime);
118
119         /**
120          * 체력바를 가져옵니다.
121          */
122         auto const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent->GetUserWidgetObject());
123
124         /**
125          * 만일 체력바를 찾았다면?
126          */
127         if (UserWidget)
128         {
129             /**
130              * 체력바에 현재 플레이어의 체력값을 표시합니다.
131              */
132             UserWidget->SetHealthBarValue((Health / MaxHealth * 100) * 0.01f);
133         }
134     }
135
136     // Called to bind functionality to input
137     void ACreature::SetupPlayerInputComponent(UInputComponent* PlayerInputComponent)
```

232 % No issues found

Error List

Entire Solution - 2 Errors 1 Warning 0 Messages Build + IntelliSense

UE4-2095 'USceneComponent::AttachTo': This function is deprecated, please use AttachToComponent instead. Please update your code to the new API before upgrading to the next release, otherwise your project will no longer compile.

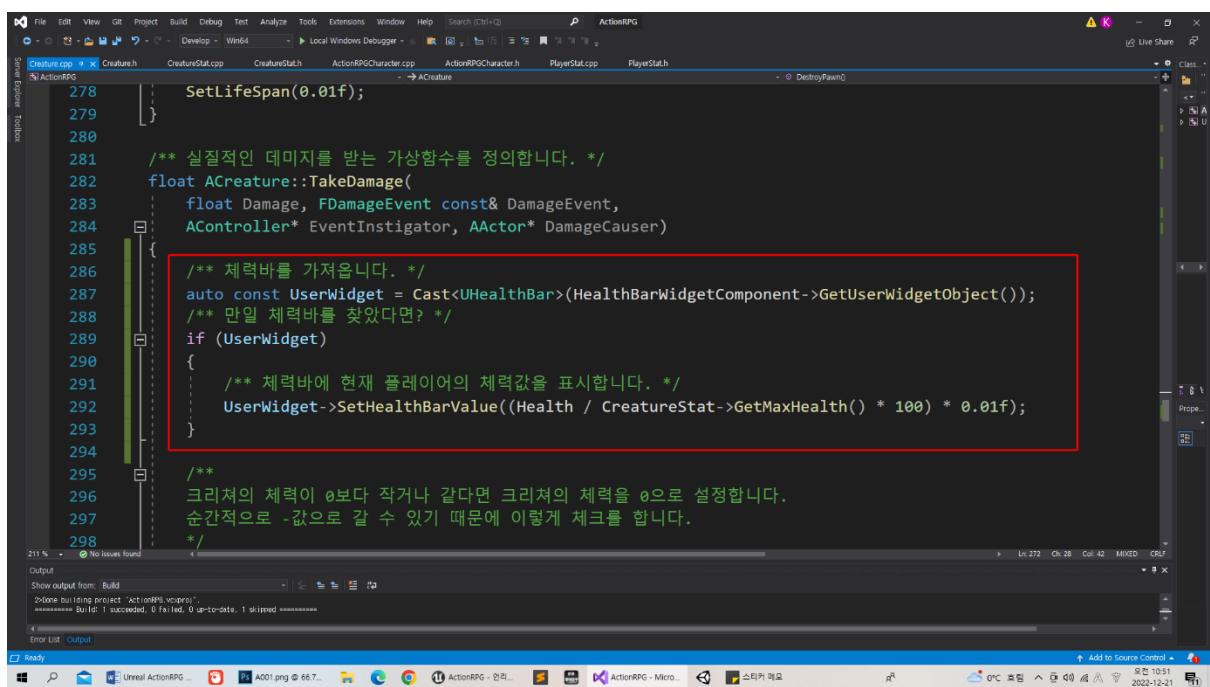
UE4-2095 'MaxHealth': undeclared identifier

Output

Build failed

Unreal ActionRPG... A001.png @ 66.7... ActionRPG - 언리... ActionRPG - Micro... 스티커 메모

0xC 흐름 오전 10:49 2022-12-21



```
278     SetLifeSpan(0.01f);
279 }
280
281 /**
282  * 실질적인 데미지를 받는 가상함수를 정의합니다.
283  */
284 float ACreature::TakeDamage(
285     float Damage, FDamageEvent const& DamageEvent,
286     AController* EventInstigator, AActor* DamageCauser)
287 {
288
289     /**
290      * 체력바를 가져옵니다.
291      */
292     auto const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent->GetUserWidgetObject());
293
294     /**
295      * 만일 체력바를 찾았다면?
296      */
297     if (UserWidget)
298     {
299
300         /**
301          * 체력바에 현재 플레이어의 체력값을 표시합니다.
302          */
303         UserWidget->SetHealthBarValue((Health / CreatureStat->GetMaxHealth() * 100) * 0.01f);
304     }
305
306     /**
307      * 크리쳐의 체력이 0보다 작거나 같다면 크리쳐의 체력을 0으로 설정합니다.
308      * 순간적으로 -값으로 갈 수 있기 때문에 이렇게 체크를 합니다.
309      */
310 }
```

211 % No issues found

Error List

Entire Solution - 2 Errors 1 Warning 0 Messages Build + IntelliSense

UE4-2095 'USceneComponent::AttachTo': This function is deprecated, please use AttachToComponent instead. Please update your code to the new API before upgrading to the next release, otherwise your project will no longer compile.

UE4-2095 'MaxHealth': undeclared identifier

Output

Show output from: Build

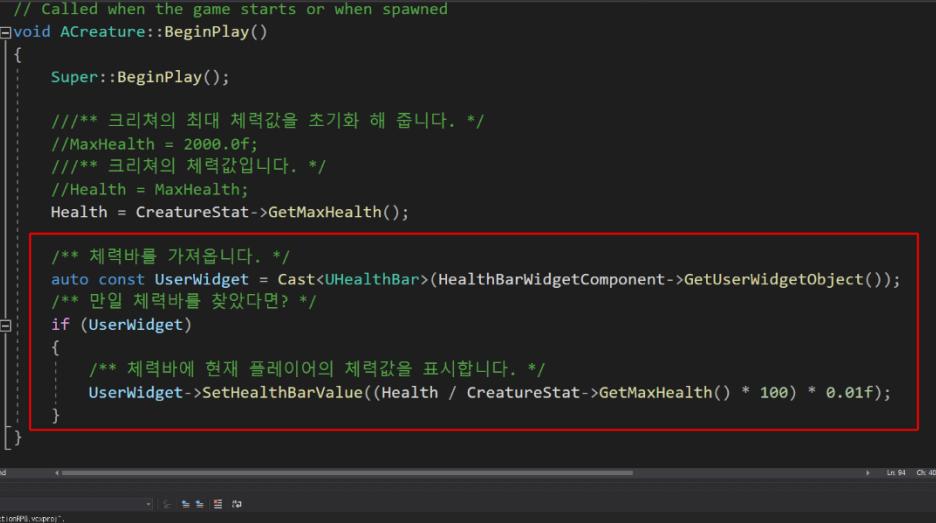
2022-12-21 10:51

Build 1 succeeded, 0 failed, 0 up-to-date, 1 skipped

Ready

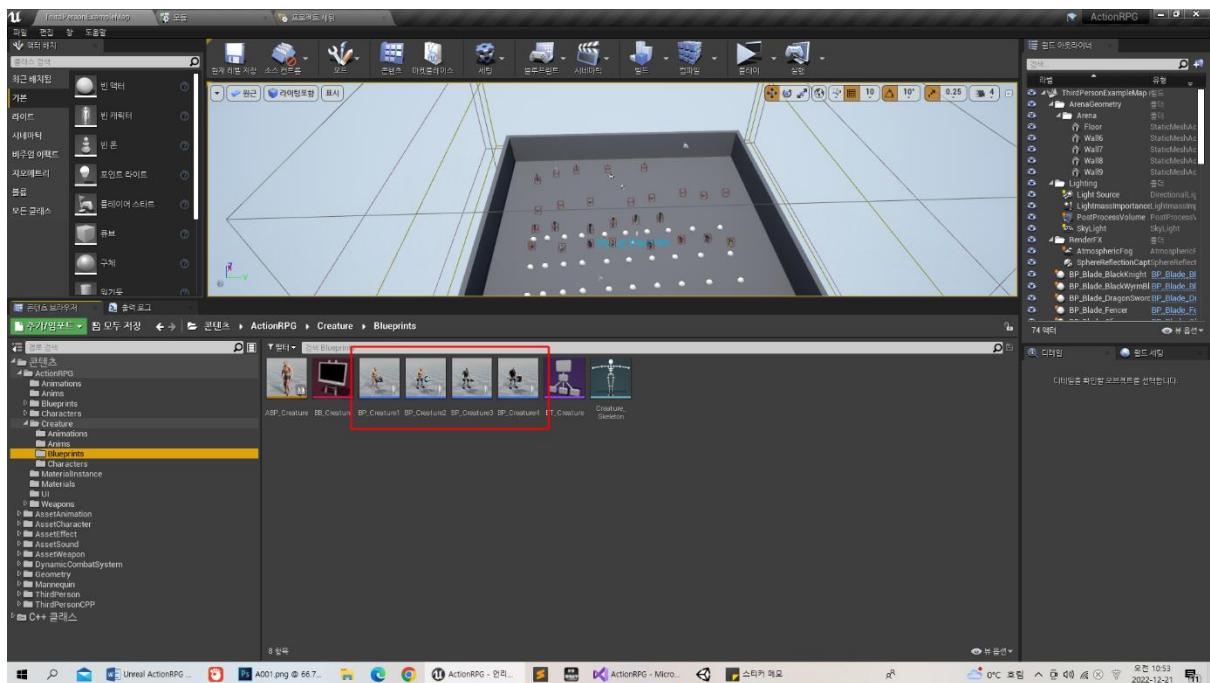
Unreal ActionRPG... A001.png @ 66.7... ActionRPG - 언리... ActionRPG - Micro... 스티커 메모

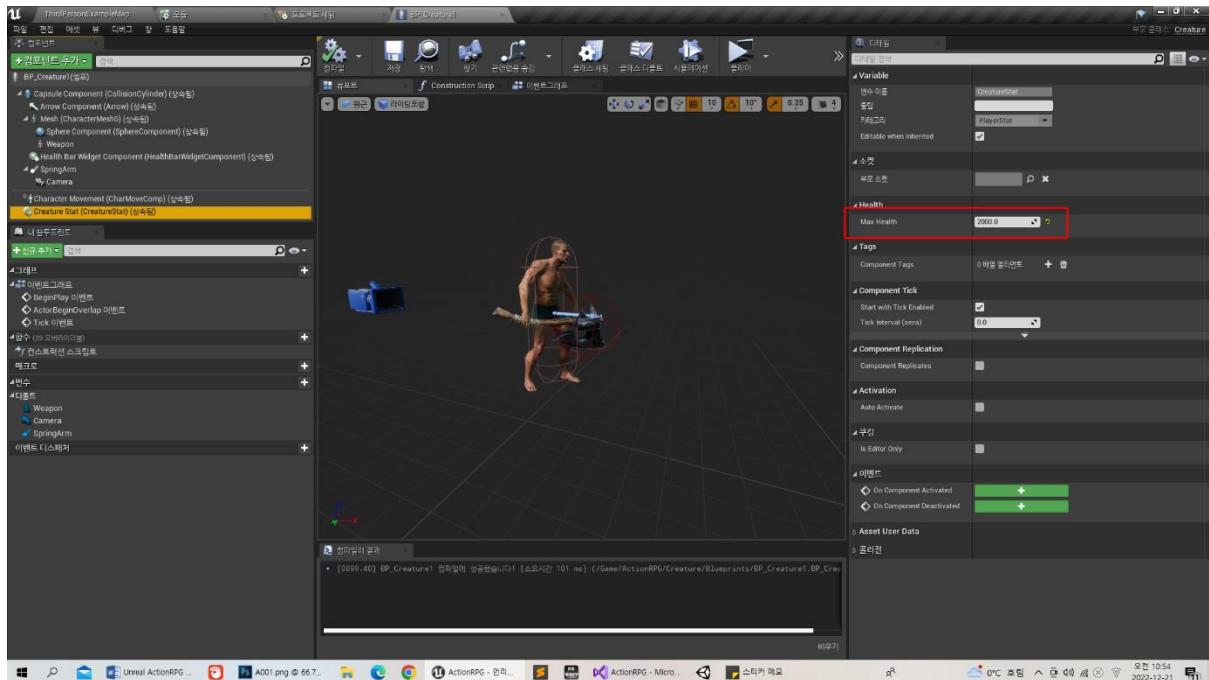
0xC 흐름 오전 10:51 2022-12-21



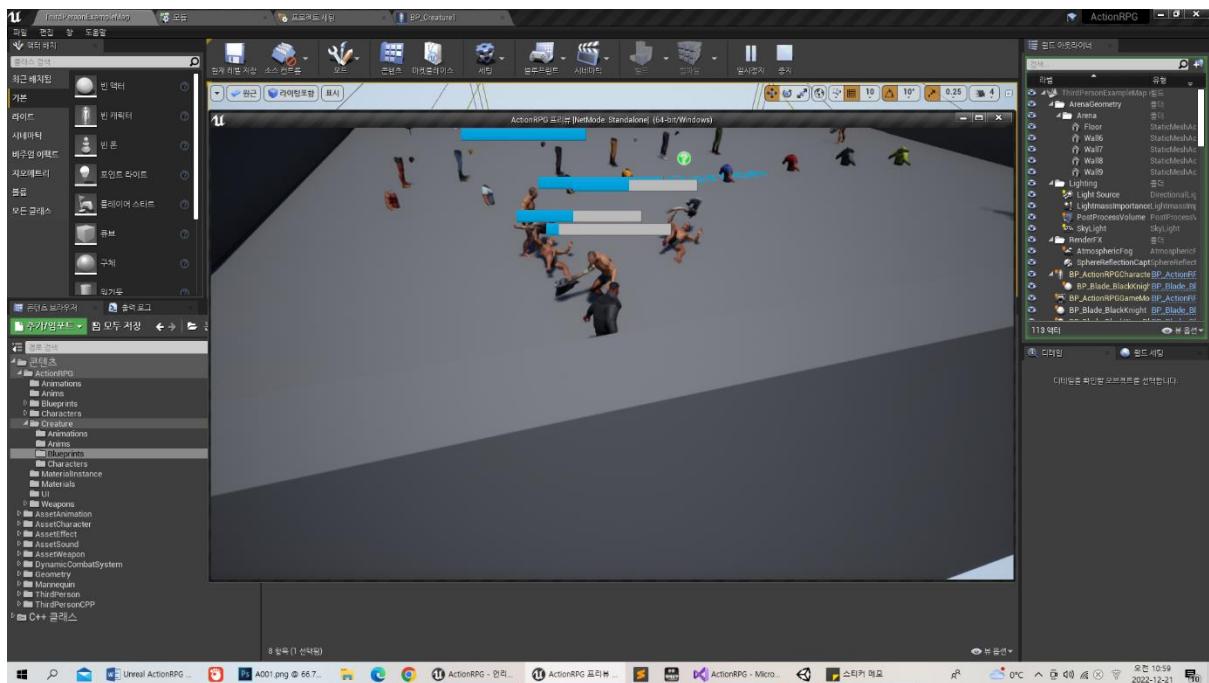
```
102 // Called when the game starts or when spawned
103 void ACreature::BeginPlay()
104 {
105     Super::BeginPlay();
106
107     /** 크리처의 최대 체력값을 초기화 해 줍니다. */
108     //MaxHealth = 2000.0f;
109     /** 크리처의 체력값입니다. */
110     //Health = MaxHealth;
111     Health = CreatureStat->GetMaxHealth();
112
113     /** 체력바를 가져옵니다. */
114     auto const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent->GetUserWidgetObject());
115     /** 만일 체력바를 찾았다면? */
116     if (UserWidget)
117     {
118         /** 체력바에 현재 플레이어의 체력값을 표시합니다. */
119         UserWidget->SetHealthBarValue((Health / CreatureStat->GetMaxHealth() * 100) * 0.01f);
120     }
121 }
122
```

크리쳐 블루프린트에서 적용해 주도록 합니다.





플레이 해서 결과를 확인합니다.



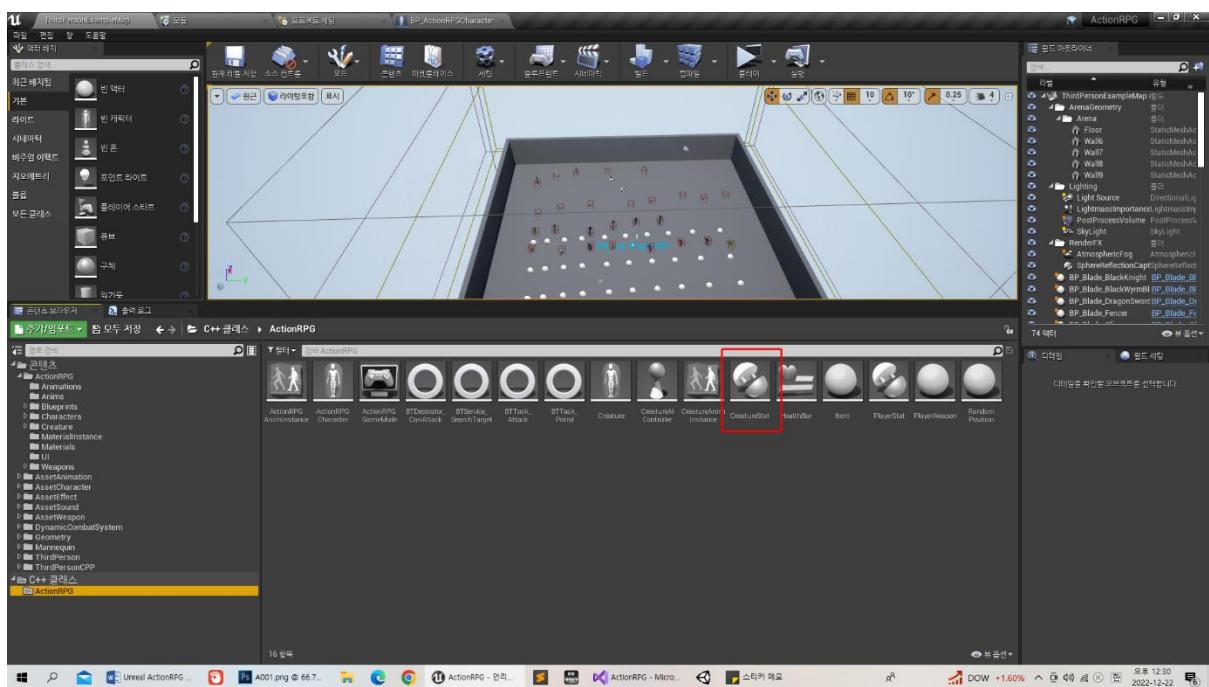
플레이어에게 주는 데미지 양을 정해 주도록 합니다.

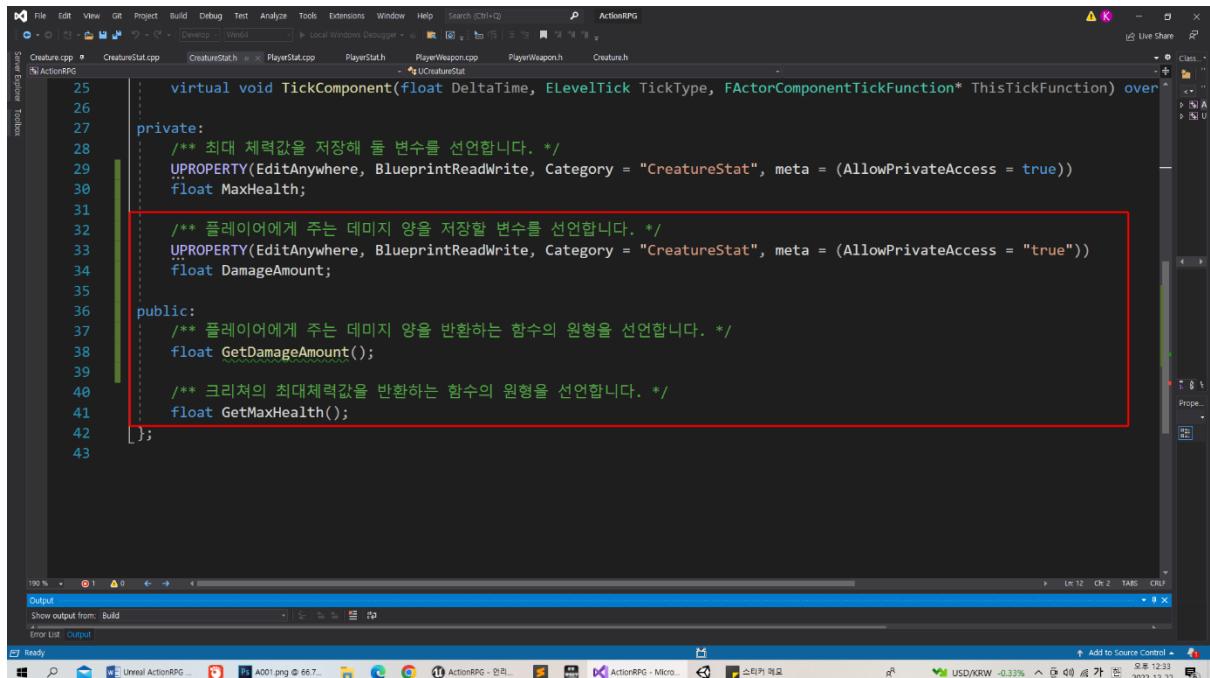
Screenshot of the Unreal Engine 4 (UE4) source code editor showing the `ActionRPG` project. The code is in `ActionRPG.cpp` and highlights a call to `UGameplayStatics::ApplyDamage` with a damage value of `100.0f`.

```

161     return nullptr;
162 }
163
164     return GameMode;
165 }
166
167     /** 충돌체크를 하는 함수를 정의합니다. */
168 void ACreature::NotifyActorBeginOverlap(AActor* OtherActor)
169 {
170     /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
171     Super::NotifyActorBeginOverlap(OtherActor);
172
173     /** 충돌한 엑터가 플레이어라면 그리고 공격중이라면? 100의 데미지를 줍니다. */
174     if (OtherActor->IsA(AActionRPGCharacter::StaticClass()) && bIsDuringAttack)
175     {
176         UGameplayStatics::ApplyDamage(OtherActor, 100.0f, nullptr, this, UDamageType::StaticClass());
177     }
178 }
179
180     /** 크리쳐가 플레이어를 향해 회전할 수 있도록 해주는 가상 함수를 정의합니다. */
181 void ACreature::FaceRotation(FRotator NewControlRotation, float DeltaTime)
182 {
183     /** 현재 크리쳐가 히저간에서 NewControlRotation 히저간으로 6초 동안 보다 빠르게 회전하면서 히저간 */

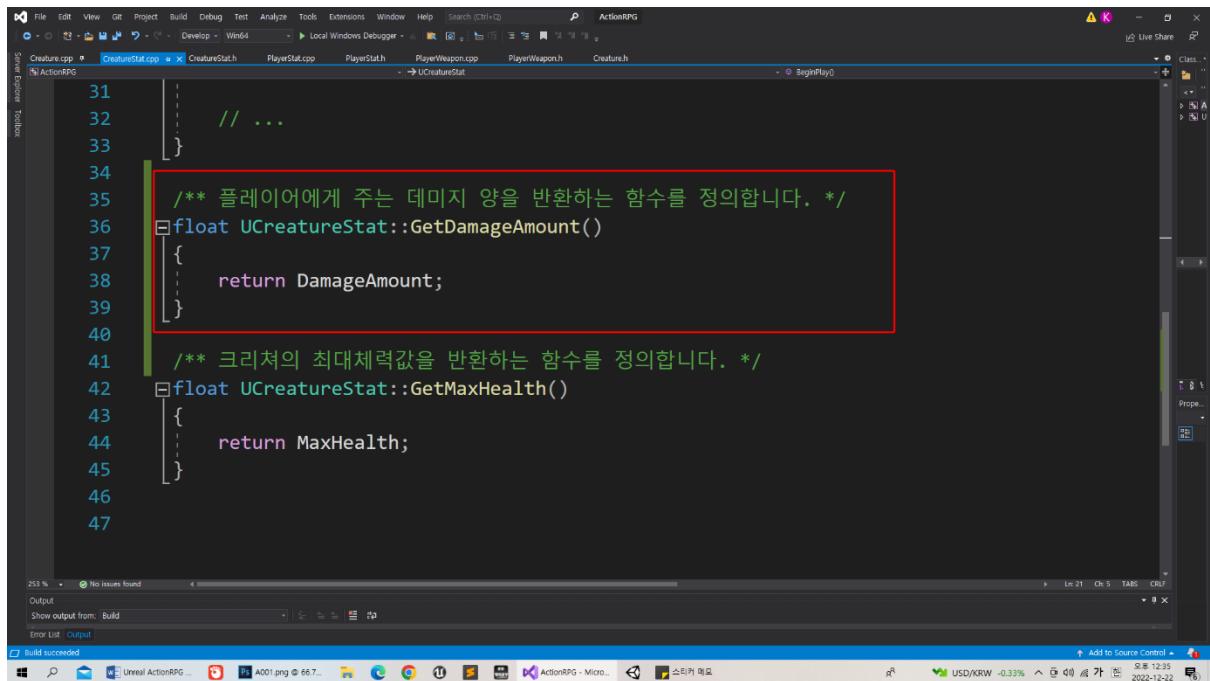
```





```
25     virtual void TickComponent(float DeltaTime, ELevelTick TickType, FActorComponentTickFunction* ThisTickFunction) override
26     {
27     }
28
29     private:
30         /** 최대 체력값을 저장해 둘 변수를 선언합니다. */
31         UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = true))
32         float MaxHealth;
33
34         /** 플레이어에게 주는 데미지 양을 저장할 변수를 선언합니다. */
35         UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = "true"))
36         float DamageAmount;
37
38     public:
39         /** 플레이어에게 주는 데미지 양을 반환하는 함수의 원형을 선언합니다. */
40         float GetDamageAmount();
41
42         /** 크리쳐의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
43         float GetMaxHealth();
44     };
45 
```

정의해 줍니다.



```
31
32     // ...
33 }
34
35     /** 플레이어에게 주는 데미지 양을 반환하는 함수를 정의합니다. */
36     float UCreatureStat::GetDamageAmount()
37     {
38         return DamageAmount;
39     }
40
41     /** 크리쳐의 최대체력값을 반환하는 함수를 정의합니다. */
42     float UCreatureStat::GetMaxHealth()
43     {
44         return MaxHealth;
45     }
46
47 
```

크리쳐 클래스에서 수정해 주도록 합니다.

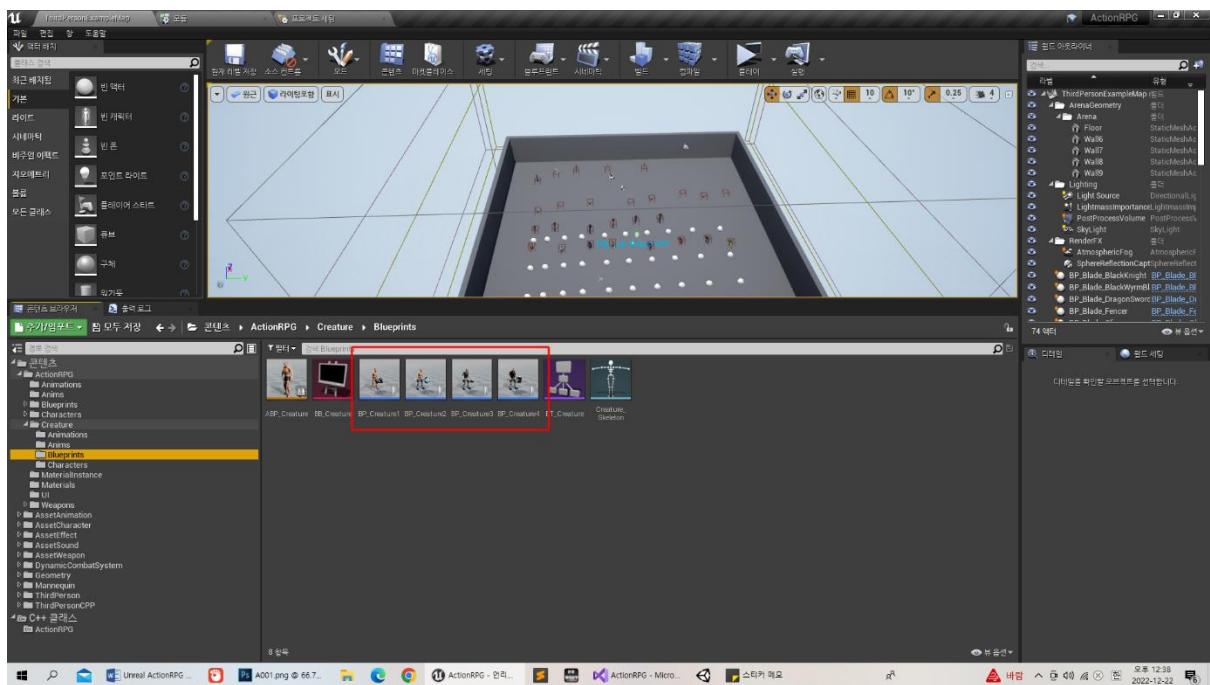
Screenshot of the Unreal Engine 4 (UE4) code editor showing the `Creature.cpp` file. The code is written in C++ and defines a creature's behavior when it begins overlapping with another actor. A specific section of the code is highlighted with a red box:

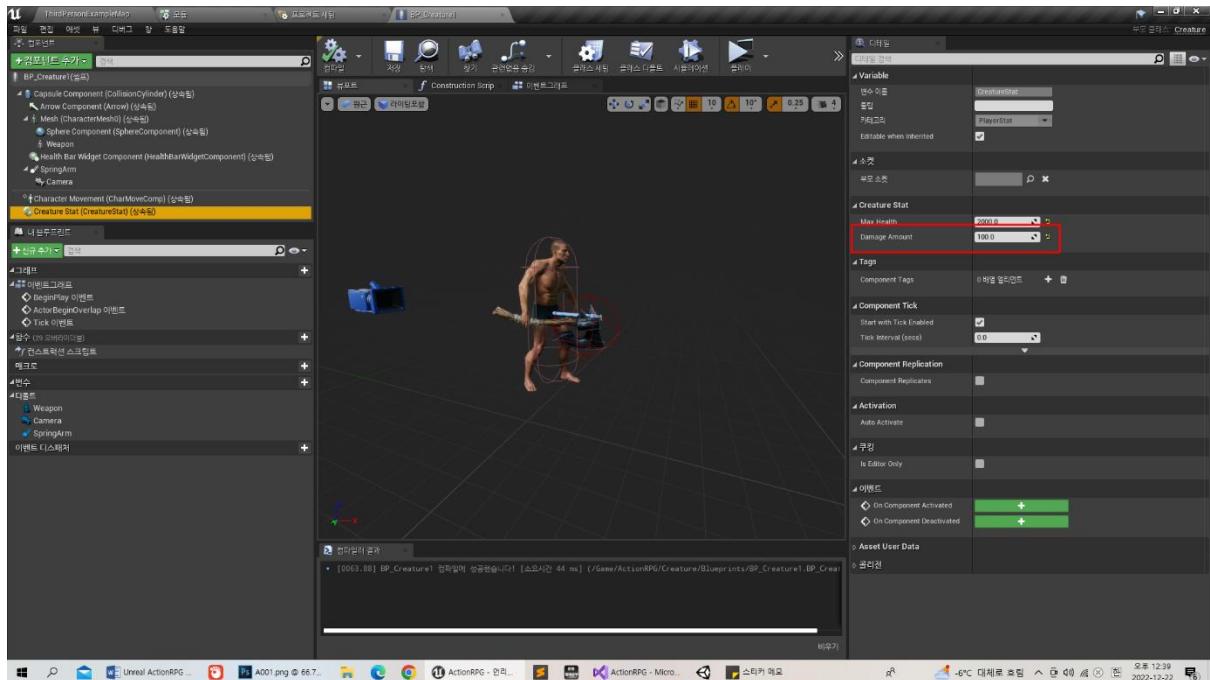
```

162     }
163
164     return GameMode;
165 }
166
167 /**
168  * 충돌체크를 하는 함수를 정의합니다.
169 */
170 void ACreature::NotifyActorBeginOverlap(AActor* OtherActor)
171 {
172     /**
173      * 부모 클래스의 함수의 내용을 모두 가져옵니다.
174      */
175     Super::NotifyActorBeginOverlap(OtherActor);
176
177     /**
178      * 충돌한 액터가 플레이어라면 그리고 공격중이라면 100의 데미지를 줍니다.
179      */
180     if (OtherActor->IsA(AActionRPGCharacter::StaticClass()) && bIsDuringAttack)
181     {
182         float DamageAmount = CreatureStat->GetDamageAmount();
183         UGameplayStatics::ApplyDamage(OtherActor, DamageAmount, nullptr, this, UDamageType::StaticClass());
184     }
185
186     /**
187      * 크리쳐가 플레이어를 향해 회전할 수 있도록 해주는 가상 함수를 정의합니다.
188      */
189     void ACreature::FaceRotation(FRotator NewControlRotation, float DeltaTime)
190     {
191         /**
192          * 현재 크리쳐의 회전값에서 NewControlRotation 회전값으로 6초 동안 부드럽게 보간하면서 회전합니다.
193          */
194     }

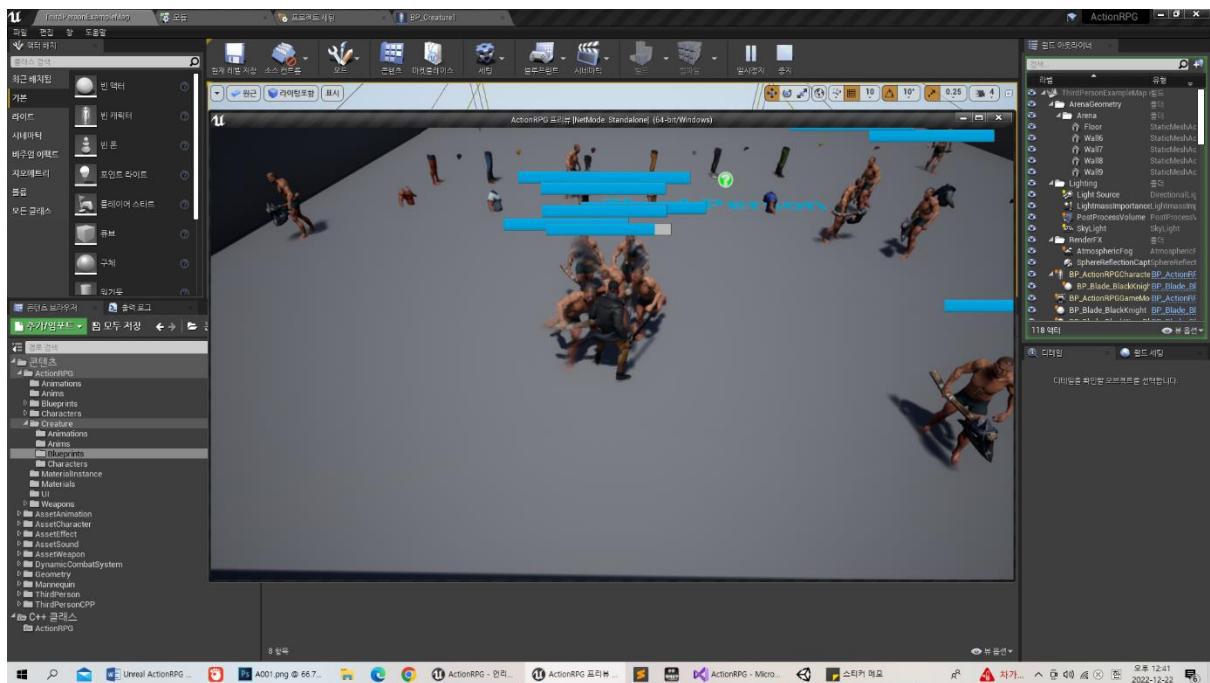
```

The highlighted code block is located between lines 177 and 184, where the creature checks if the overlapping actor is a player and if it's in attack mode, then applies damage to the player.

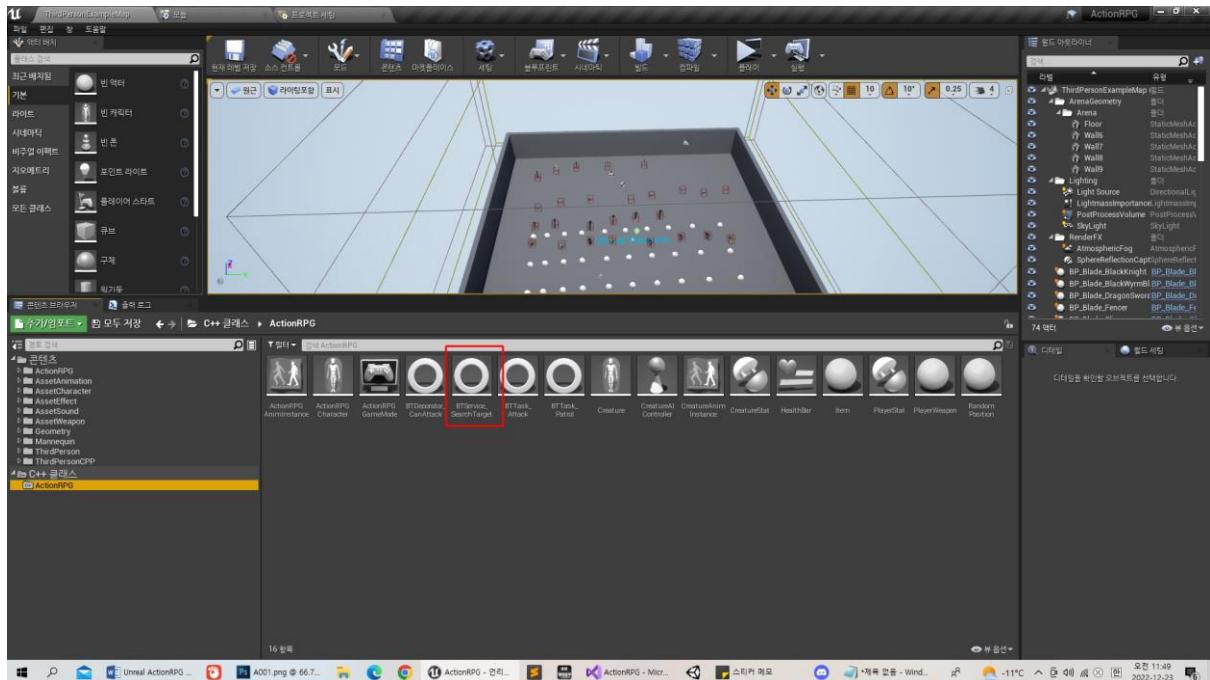




플레이를 해서 결과를 확인합니다.



BTService_SearchTarget의 하드코딩된 부분을 수정해 주도록 합니다.



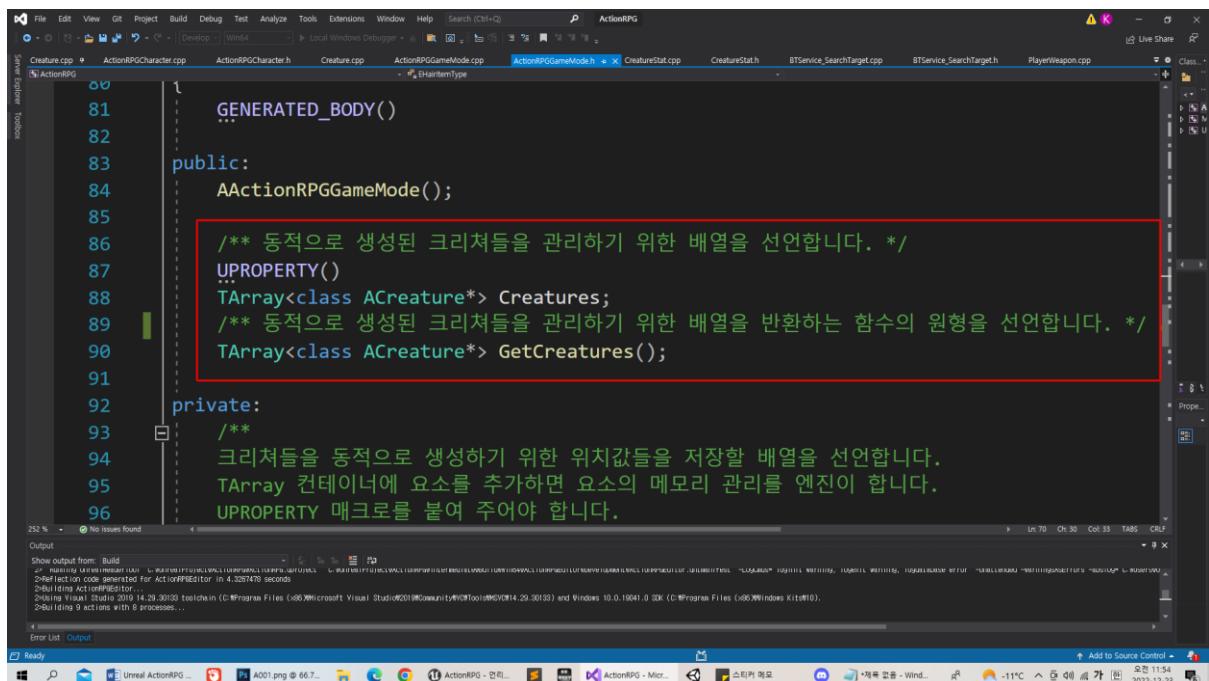
```

1 31 	/** 플레이어에게 주는 데미지 양을 저장할 변수를 선언합니다. */
2 32 	UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = "true"))
3 33 	float DamageAmount;
4
5 34
6 35 	/** 플레이어를 찾는 반경을 저장할 변수를 선언합니다. */
7 36 	UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = "true"))
8 37 	float SearchRadius;
9
10 38
11 39
12 40 	public:
13 	/** 플레이어를 찾는 반경을 반환하는 함수의 원형을 선언합니다. */
14 	/** float GetSearchRadius(); */
15
16
17 	/** 플레이어에게 주는 데미지 양을 반환하는 함수의 원형을 선언합니다. */
18 	/** float GetDamageAmount(); */
19
20 	/** 크리쳐의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
21 	/** float GetMaxHealth(); */
22
23 };

24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50

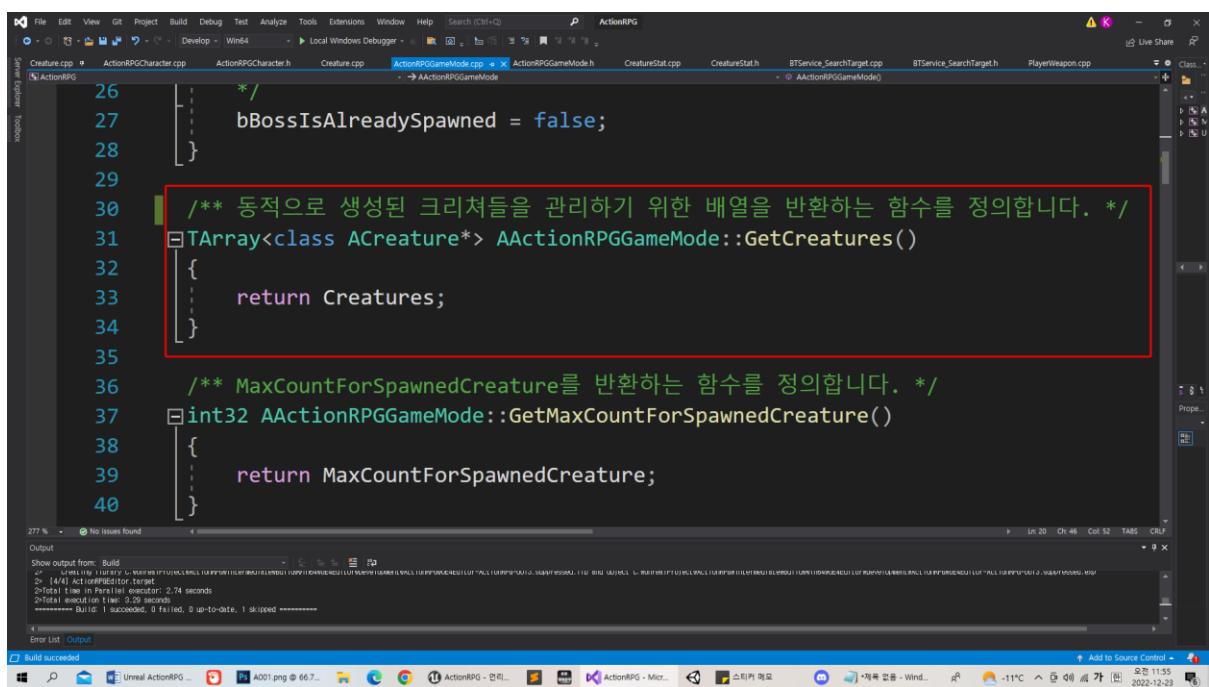
```

The code block shows a C++ class definition for 'CreatureStat'. It includes properties for 'DamageAmount' and 'SearchRadius', and methods for 'GetSearchRadius', 'GetDamageAmount', and 'GetMaxHealth'. A red box highlights the code block.



```
81     GENERATED_BODY()
82
83     public:
84         AACTIONRPGGAMEMODE();
85
86         /** 동적으로 생성된 크리쳐들을 관리하기 위한 배열을 선언합니다. */
87         UPROPERTY()
88         TArray<class ACreature*> Creatures;
89         /** 동적으로 생성된 크리쳐들을 관리하기 위한 배열을 반환하는 함수의 원형을 선언합니다. */
90         TArray<class ACreature*> GetCreatures();
91
92     private:
93         /**
94         * 크리쳐들을 동적으로 생성하기 위한 위치값들을 저장할 배열을 선언합니다.
95         * TArray 컨테이너에 요소를 추가하면 요소의 메모리 관리를 엔진이 합니다.
96         * UPROPERTY 매크로를 붙여 주어야 합니다.

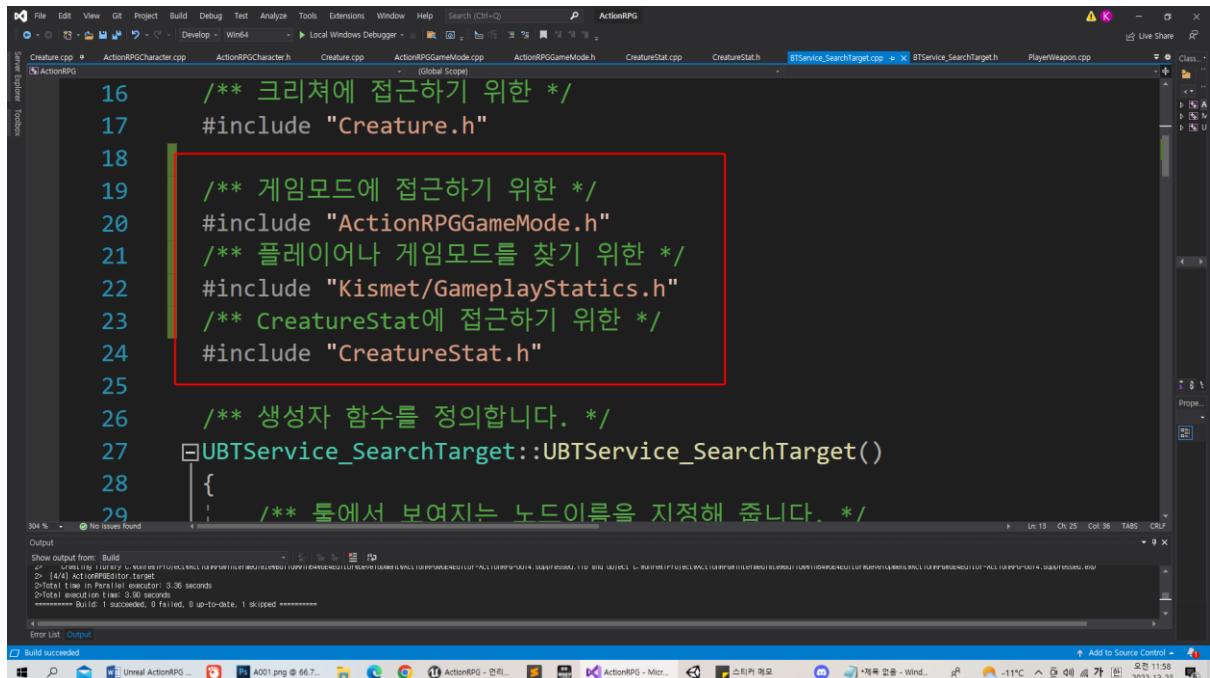
```



```
26     */
27     bBossIsAlreadySpawned = false;
28 }
29
30         /** 동적으로 생성된 크리쳐들을 관리하기 위한 배열을 반환하는 함수를 정의합니다. */
31         TArray<class ACreature*> AACTIONRPGGAMEMODE::GetCreatures()
32     {
33         return Creatures;
34     }
35
36         /** MaxCountForSpawnedCreature를 반환하는 함수를 정의합니다. */
37         int32 AACTIONRPGGAMEMODE::GetMaxCountForSpawnedCreature()
38     {
39         return MaxCountForSpawnedCreature;
40     }

```

BTService_SearchTarget 클래스에서 수정해 주도록 합니다.

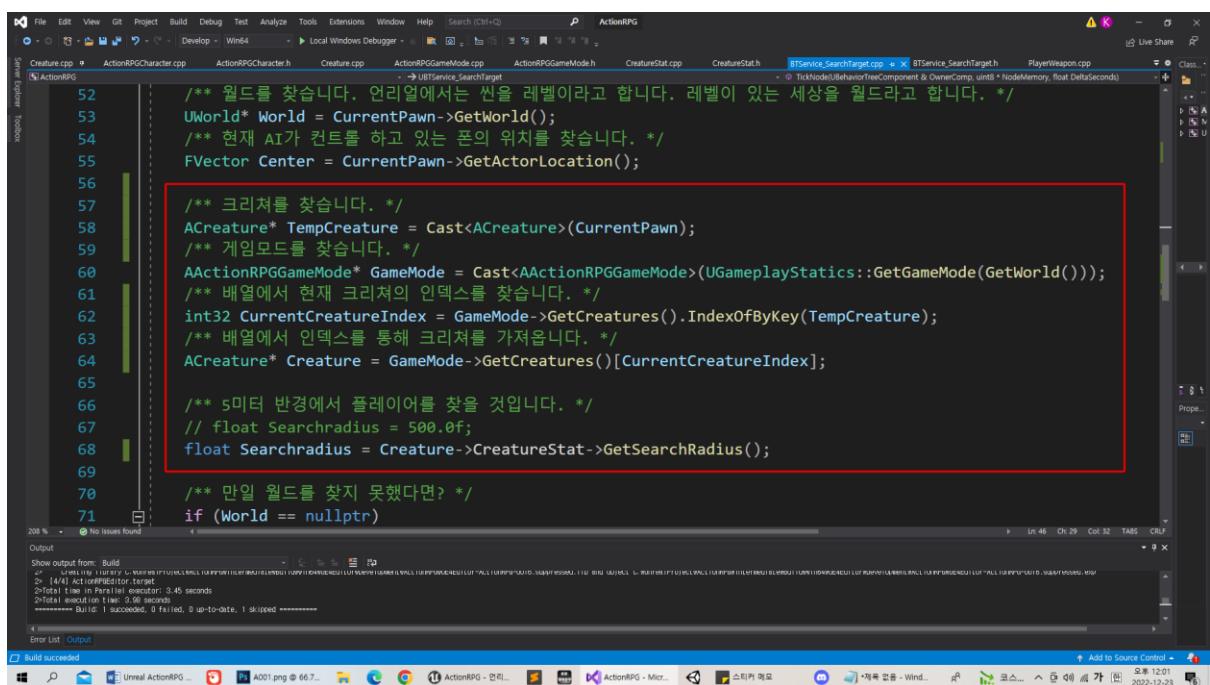


```
16     /** 크리쳐에 접근하기 위한 */
17     #include "Creature.h"
18
19     /** 게임모드에 접근하기 위한 */
20     #include "ActionRPGGameMode.h"
21     /** 플레이어나 게임모드를 찾기 위한 */
22     #include "Kismet/GameplayStatics.h"
23     /** CreatureStat에 접근하기 위한 */
24     #include "CreatureStat.h"
25
26     /** 생성자 함수를 정의합니다. */
27     UBTService_SearchTarget::UBTService_SearchTarget()
28     {
29         /** 둘에서 보여지는 노드이름을 지정해 줍니다. */
```

Output:
Build output from: Build
1/4/41 ActionRPGEditor.Target
2/ Total time in Parallel executor: 3.46 seconds
3/ Total execution time: 3.46 seconds
***** Build 0 succeeded, 0 failed, 0 up-to-date, 1 skipped *****

Error List: Output

Build succeeded



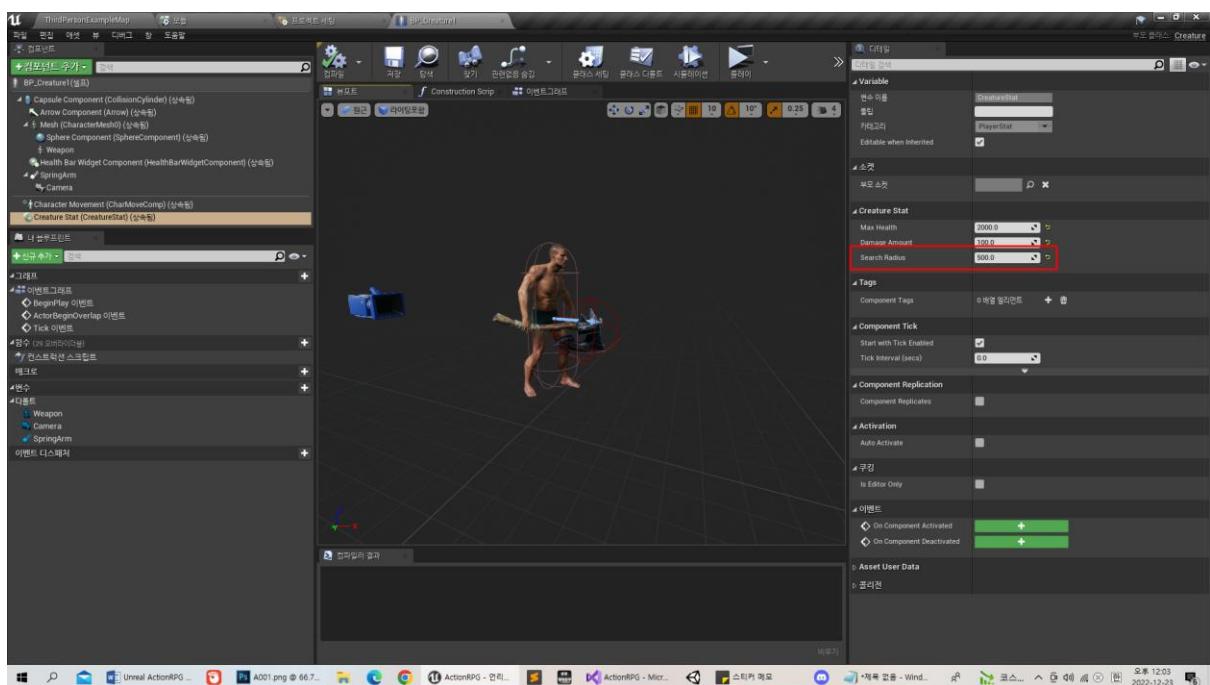
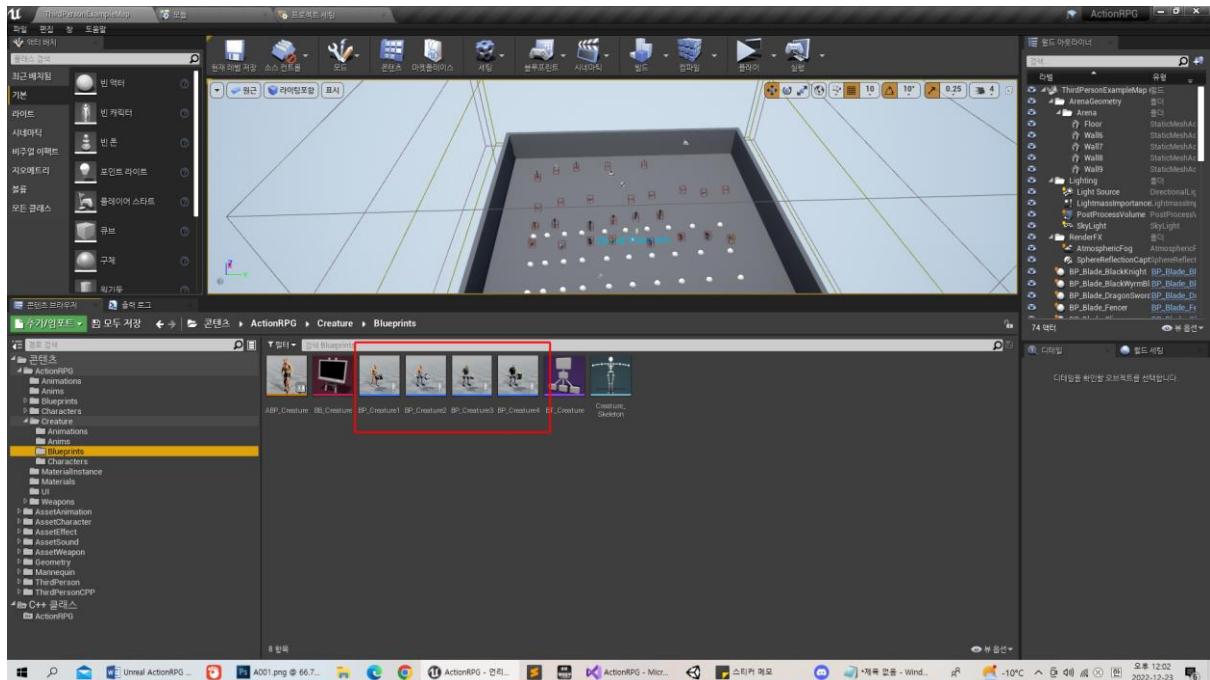
```
52     /** 월드를 찾습니다. 언리얼에서는 씬을 레벨이라고 합니다. 레벨이 있는 세상을 월드라고 합니다. */
53     UWorld* World = CurrentPawn->GetWorld();
54     /** 현재 AI가 컨트롤 하고 있는 푸니 위치를 찾습니다. */
55     FVector Center = CurrentPawn->GetActorLocation();
56
57     /** 크리쳐를 찾습니다. */
58     ACreature* TempCreature = Cast<ACreature>(CurrentPawn);
59     /** 게임모드를 찾습니다. */
60     AAActionRPGGameMode* GameMode = Cast<AAActionRPGGameMode>(UGameplayStatics::GetGameMode(GetWorld()));
61     /** 배열에서 현재 크리쳐의 인덱스를 찾습니다. */
62     int32 CurrentCreatureIndex = GameMode->GetCreatures().IndexOfByKey(TempCreature);
63     /** 배열에서 인덱스를 통해 크리쳐를 가져옵니다. */
64     ACreature* Creature = GameMode->GetCreatures()[CurrentCreatureIndex];
65
66     /** 5미터 반경에서 플레이어를 찾을 것입니다. */
67     // float Searchradius = 500.0f;
68     float Searchradius = Creature->CreatureStat->GetSearchRadius();
69
70     /** 만일 월드를 찾지 못했다면? */
71     if (World == nullptr)
```

Output:
Build output from: Build
1/4/41 ActionRPGEditor.Target
2/ Total time in Parallel executor: 3.46 seconds
3/ Total execution time: 3.46 seconds
***** Build 0 succeeded, 0 failed, 0 up-to-date, 1 skipped *****

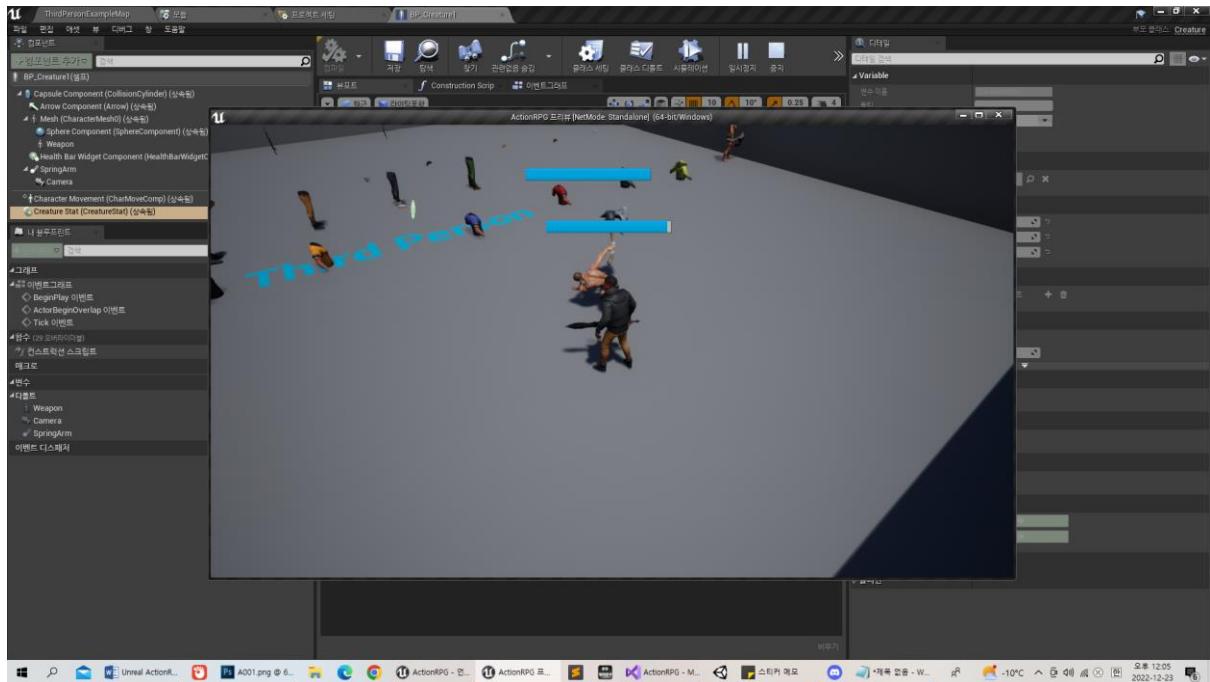
Error List: Output

Build succeeded

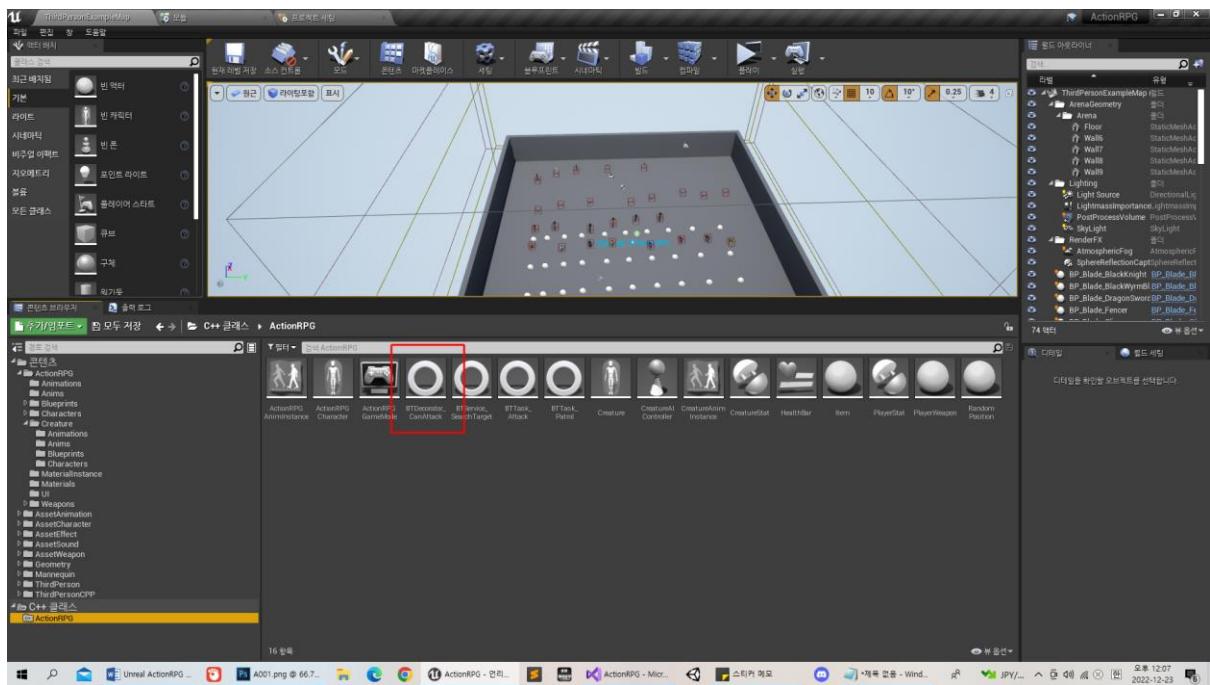
크리쳐 블루프린트에서 적용해 줍니다.

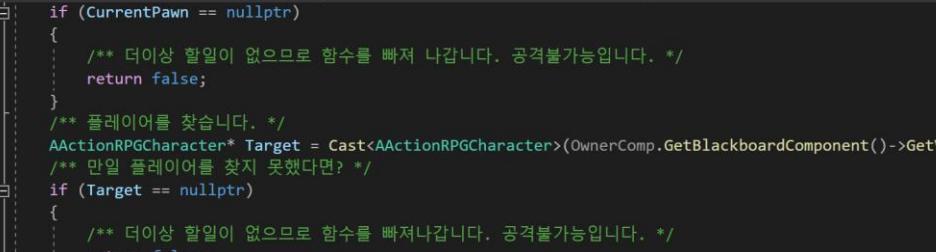


플레이를 해서 결과를 확인합니다.

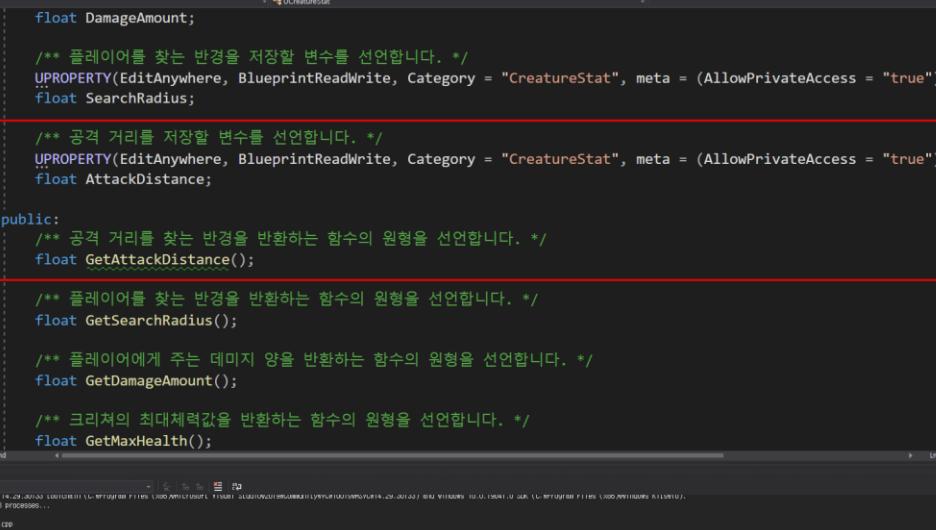


BTDecorator_CanAttack 클래스의 하드코딩된 부분을 수정해 주도록 합니다.



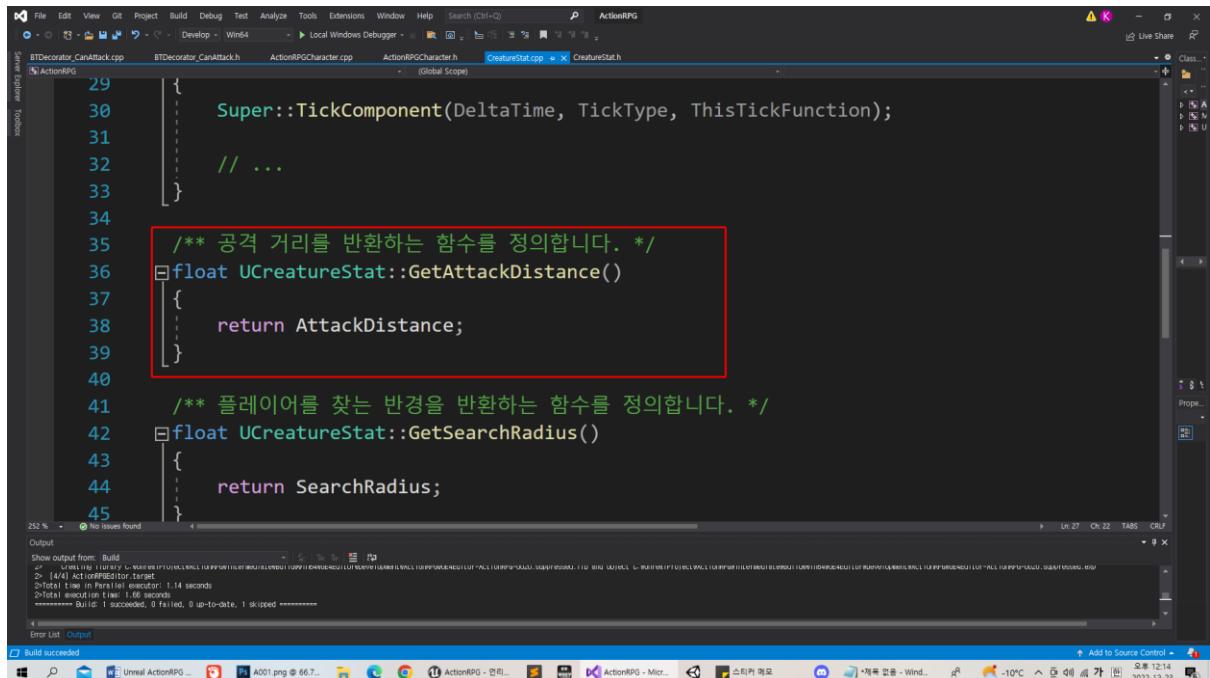


```
28     if (CurrentPawn == nullptr)
29     {
30         /** 더이상 할일이 없으므로 함수를 빠져 나갑니다. 공격불가능입니다. */
31         return false;
32     }
33     /** 플레이어를 찾습니다. */
34     AACTIONRPGCHARACTER* Target = Cast<AACTIONRPGCHARACTER>(OwnerComp.GetBlackboardComponent()->GetValueAsObject("Target"));
35     /** 만일 플레이어를 찾지 못했다면? */
36     if (Target == nullptr)
37     {
38         /** 더이상 할일이 없으므로 함수를 빠져나갑니다. 공격불가능입니다. */
39         return false;
40     }
41     /** 플레이어와의 거리가 공격거리인 2미터 보다 작거나 같은지 리턴합니다. */
42     return bResult && Target->GetDistanceTo(CurrentPawn) <= 200.0f;
43 }
```



```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) ActionRPG
BTDecorator_CanAttack.cpp BTDecorator_CanAttack.h ActionRPGCharacter.cpp ActionRPGCharacter.h CreatureStat.cpp CreatureStat.h
ActionRPG
34     float DamageAmount;
35
36     /** 플레이어를 찾는 반경을 저장할 변수를 선언합니다. */
37     UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = "true"))
38     float SearchRadius;
39
40     /** 공격 거리를 저장할 변수를 선언합니다. */
41     UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = "true"))
42     float AttackDistance;
43
44 public:
45     /** 공격 거리를 찾는 반경을 반환하는 함수의 원형을 선언합니다. */
46     float GetAttackDistance();
47
48     /** 플레이어를 찾는 반경을 반환하는 함수의 원형을 선언합니다. */
49     float GetSearchRadius();
50
51     /** 플레이어에게 주는 데미지 양을 반환하는 함수의 원형을 선언합니다. */
52     float GetDamageAmount();
53
54     /** 크리쳐의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
55     float GetMaxHealth();
```

구현부에서 정의해 줍니다.

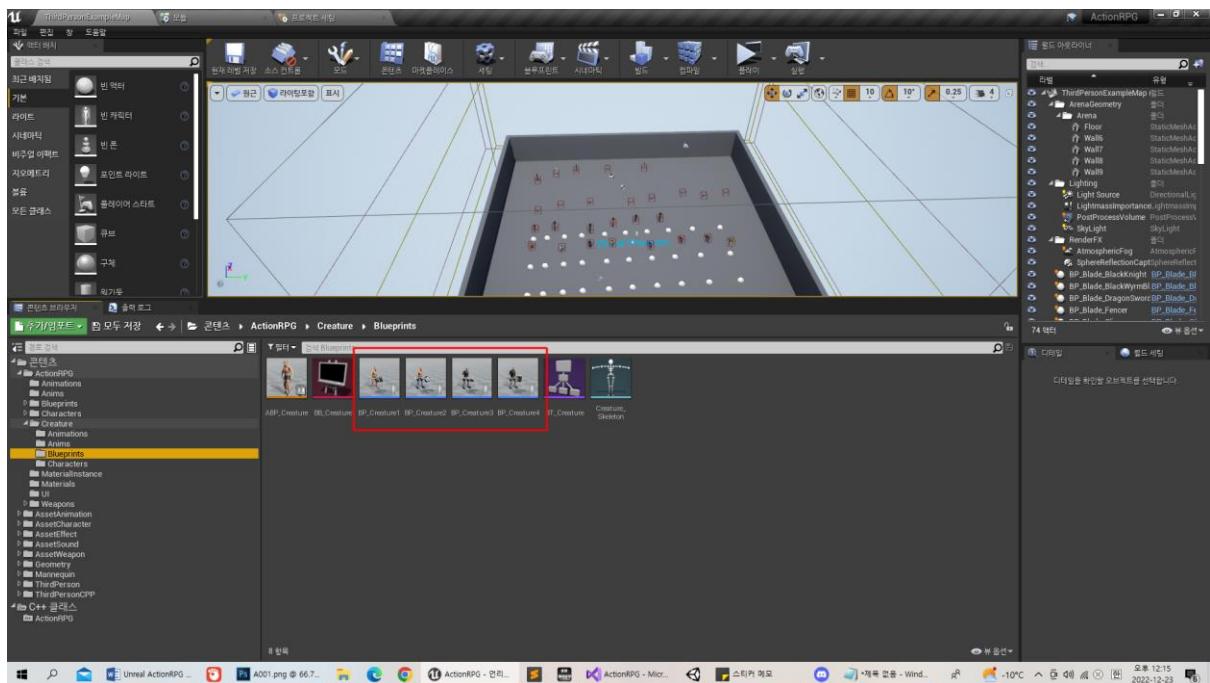


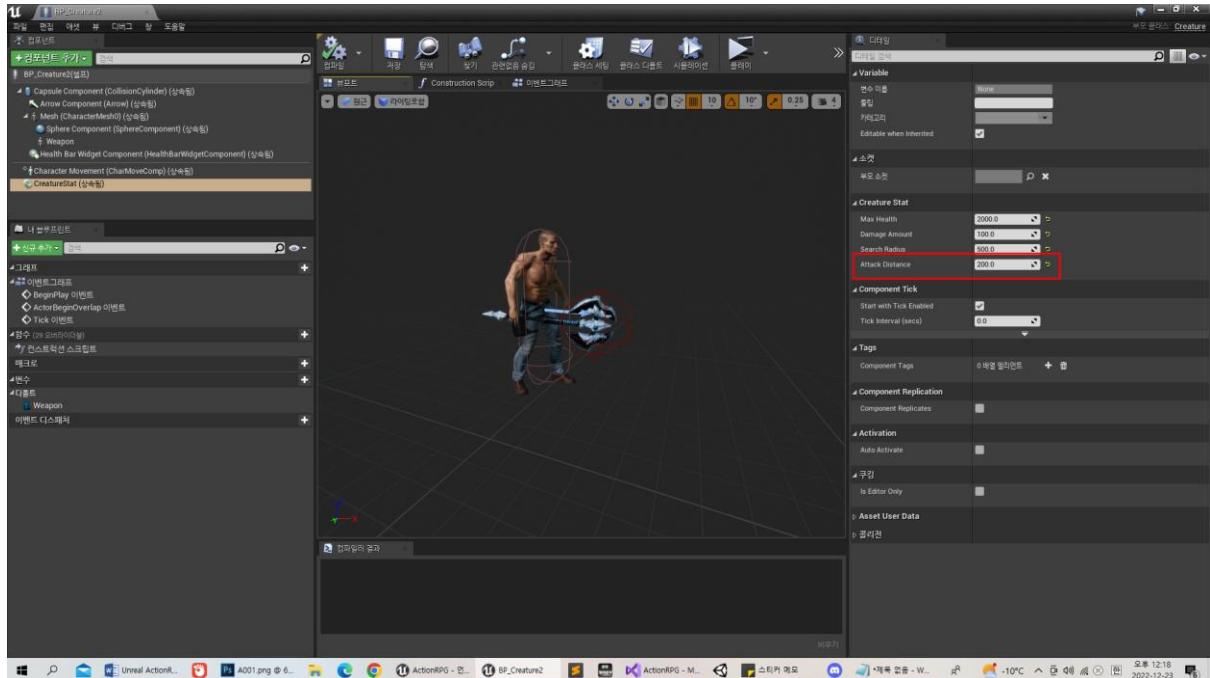
```

29     Super::TickComponent(DeltaTime, TickType, ThisTickFunction);
30
31     // ...
32
33 }
34
35     /** 공격 거리를 반환하는 함수를 정의합니다. */
36 float UCreatureStat::GetAttackDistance()
37 {
38     return AttackDistance;
39 }
40
41     /** 플레이어를 찾는 반경을 반환하는 함수를 정의합니다. */
42 float UCreatureStat::GetSearchRadius()
43 {
44     return SearchRadius;
45 }

```

크리쳐 블루프린트에서 적용해 주도록 합니다.





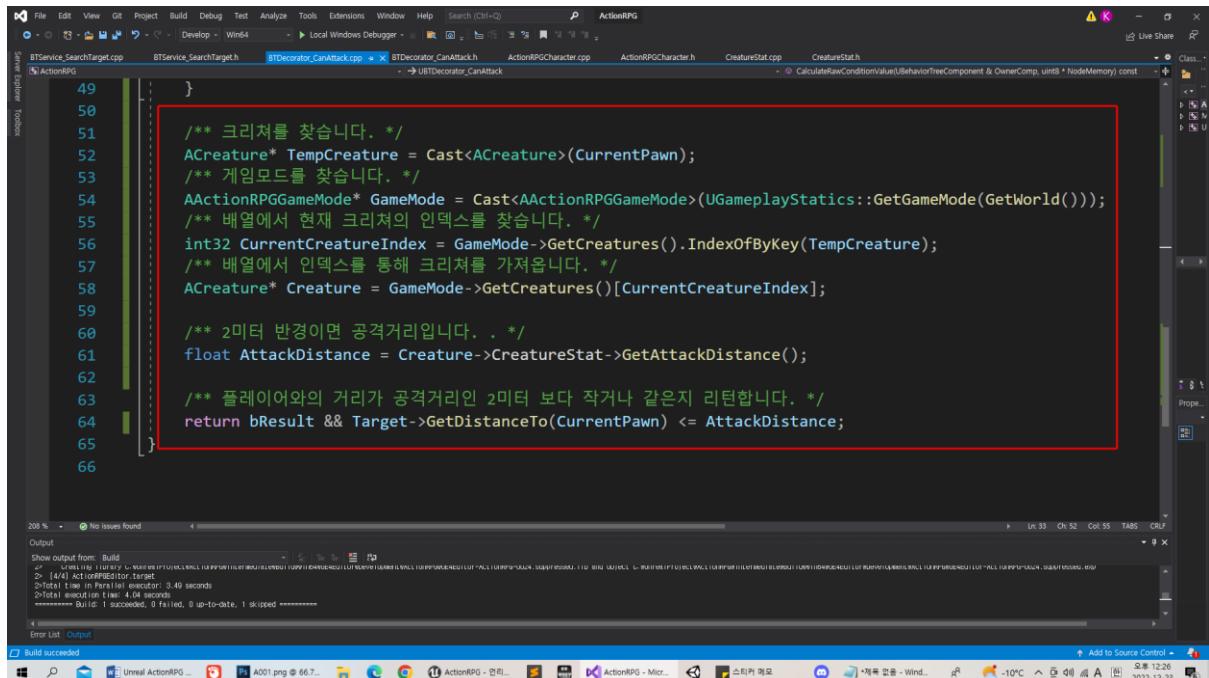
BTDecorator_CanAttack 클래스에서 수정해 줍니다.

```

File Edit View GI Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+F) ActionRPG
BTService_SearchTarget.cpp BTService_SearchTarget.h BTDecorator_CanAttack.cpp ActionRPGCharacter.cpp ActionRPGCharacter.h CreatureStat.cpp CreatureStat.h
ActionRPG
10  /** 블랙보드컴포넌트에 접근하기 위한 */
11  #include "BehaviorTree/BlackboardComponent.h"
12
13  /** 게임모드에 접근하기 위한 */
14  #include "ActionRPGGameMode.h"
15  /** 플레이어나 게임모드를 찾기 위한 */
16  #include "Kismet/GameplayStatics.h"
17  /** CreatureStat에 접근하기 위한 */
18  #include "CreatureStat.h"
19  /** 크리쳐에 접근하기 위한 */
20  #include "Creature.h"
21
22  /** 생성자 함수를 정의합니다. */
23  UBTDecorator_CanAttack::UBTDecorator_CanAttack()
24  {

```

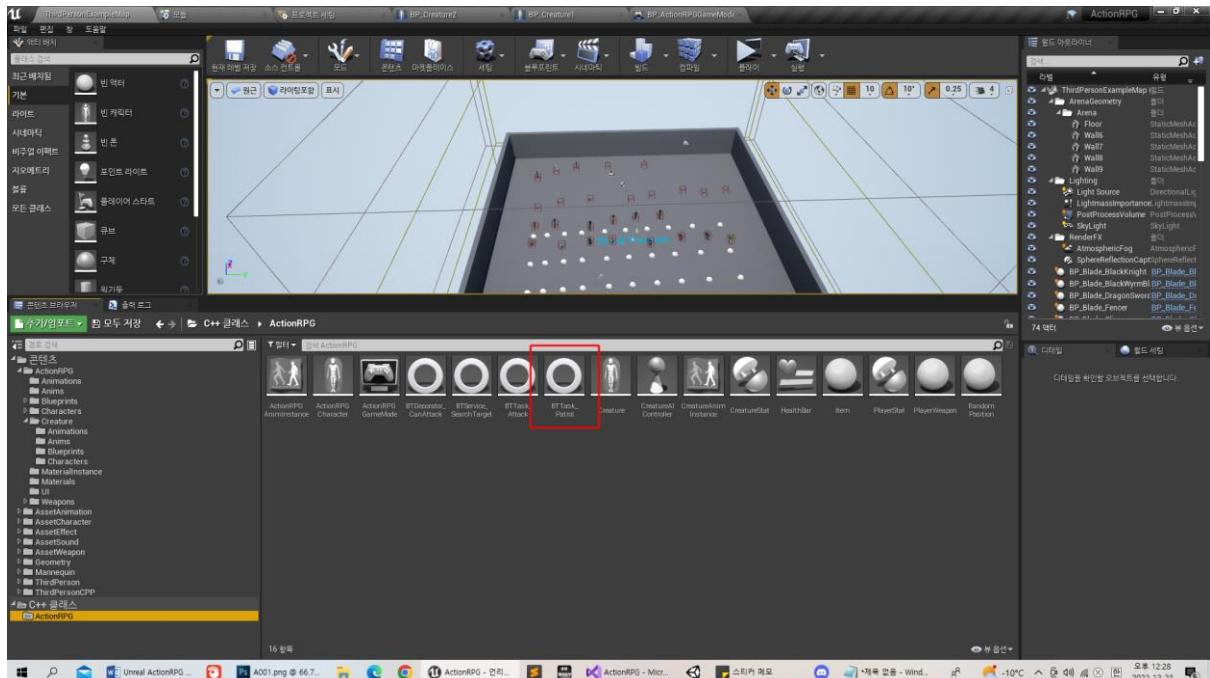
The code block shows the implementation of the `UBTDecorator_CanAttack` constructor. The first 20 lines are part of the class definition and include necessary headers for the BlackboardComponent, ActionRPGGameMode, GameplayStatics, CreatureStat, and Creature classes. The last four lines define the constructor itself, which is highlighted with a red box.



```

49 }
50 }
51 /**
52 * 크리쳐를 찾습니다.
53 * ACreature* TempCreature = Cast<ACreature>(CurrentPawn);
54 /**
55 * 게임모드를 찾습니다.
56 * AActionRPGGameMode* GameMode = Cast<AActionRPGGameMode>(UGameplayStatics::GetGameMode(GetWorld()));
57 /**
58 * 배열에서 현재 크리쳐의 인덱스를 찾습니다.
59 * int32 CurrentCreatureIndex = GameMode->GetCreatures().IndexOfByKey(TempCreature);
60 /**
61 * 배열에서 인덱스를 통해 크리쳐를 가져옵니다.
62 * ACreature* Creature = GameMode->GetCreatures()[CurrentCreatureIndex];
63 /**
64 * 2미터 반경이면 공격거리입니다.
65 * float AttackDistance = Creature->CreatureStat->GetAttackDistance();
66 */
67
68 /**
69 * 플레이어와의 거리가 공격거리인 2미터 보다 작거나 같은지 리턴합니다.
70 */
71 return bResult && Target->GetDistanceTo(CurrentPawn) <= AttackDistance;
72
73 }
74
75 }
76
77 }
78
79 }
80
81 }
82
83 }
84
85 }
86
87 }
88
89 }
90
91 }
92
93 }
94
95 }
96
97 }
98
99 }
100
101 }
102
103 }
104
105 }
106
107 }
108
109 }
110
111 }
112
113 }
114
115 }
116
117 }
118
119 }
120
121 }
122
123 }
124
125 }
126
127 }
128
129 }
130
131 }
132
133 }
134
135 }
136
137 }
138
139 }
140
141 }
142
143 }
144
145 }
146
147 }
148
149 }
150
151 }
152
153 }
154
155 }
156
157 }
158
159 }
160
161 }
162
163 }
164
165 }
166
167 }
168
169 }
170
171 }
172
173 }
174
175 }
176
177 }
178
179 }
180
181 }
182
183 }
184
185 }
186
187 }
188
189 }
190
191 }
192
193 }
194
195 }
196
197 }
198
199 }
200
201 }
202
203 }
204
205 }
206
207 }
208
209 }
210
211 }
212
213 }
214
215 }
216
217 }
218
219 }
220
221 }
222
223 }
224
225 }
226
227 }
228
229 }
230
231 }
232
233 }
234
235 }
236
237 }
238
239 }
240
241 }
242
243 }
244
245 }
246
247 }
248
249 }
250
251 }
252
253 }
254
255 }
256
257 }
258
259 }
259
260 }
261
262 }
263
264 }
265
266 }
267
268 }
269
270 }
271
272 }
273
274 }
275
276 }
277
278 }
279
280 }
281
282 }
283
284 }
285
286 }
287
288 }
289
290 }
291
292 }
293
294 }
295
296 }
297
298 }
299
299
300 }
301
302 }
303
304 }
305
306 }
307
308 }
309
309
310 }
311
312 }
313
314 }
315
316 }
317
318 }
319
319
320 }
321
322 }
323
324 }
325
326 }
327
328 }
329
329
330 }
331
332 }
333
334 }
335
336 }
337
338 }
339
339
340 }
341
342 }
343
344 }
345
346 }
347
348 }
349
349
350 }
351
352 }
353
354 }
355
356 }
357
358 }
359
359
360 }
361
362 }
363
364 }
365
366 }
367
368 }
369
369
370 }
371
372 }
373
374 }
375
376 }
377
378 }
379
379
380 }
381
382 }
383
384 }
385
386 }
387
388 }
389
389
390 }
391
392 }
393
394 }
395
396 }
397
398 }
399
399
400 }
401
402 }
403
404 }
405
406 }
407
408 }
409
409
410 }
411
412 }
413
414 }
415
416 }
417
418 }
419
419
420 }
421
422 }
423
424 }
425
426 }
427
428 }
429
429
430 }
431
432 }
433
434 }
435
436 }
437
438 }
439
439
440 }
441
442 }
443
444 }
445
446 }
447
448 }
449
449
450 }
451
452 }
453
454 }
455
456 }
457
458 }
459
459
460 }
461
462 }
463
464 }
465
466 }
467
468 }
469
469
470 }
471
472 }
473
474 }
475
476 }
477
478 }
479
479
480 }
481
482 }
483
484 }
485
486 }
487
488 }
489
489
490 }
491
492 }
493
494 }
495
496 }
497
498 }
499
499
500 }
501
502 }
503
504 }
505
506 }
507
508 }
509
509
510 }
511
512 }
513
514 }
515
516 }
517
518 }
519
519
520 }
521
522 }
523
524 }
525
526 }
527
528 }
529
529
530 }
531
532 }
533
534 }
535
536 }
537
538 }
539
539
540 }
541
542 }
543
544 }
545
546 }
547
548 }
549
549
550 }
551
552 }
553
554 }
555
556 }
557
558 }
559
559
560 }
561
562 }
563
564 }
565
566 }
567
568 }
569
569
570 }
571
572 }
573
574 }
575
576 }
577
578 }
579
579
580 }
581
582 }
583
584 }
585
586 }
587
588 }
589
589
590 }
591
592 }
593
594 }
595
596 }
597
598 }
599
599
600 }
601
602 }
603
604 }
605
606 }
607
608 }
609
609
610 }
611
612 }
613
614 }
615
616 }
617
618 }
619
619
620 }
621
622 }
623
624 }
625
626 }
627
628 }
629
629
630 }
631
632 }
633
634 }
635
636 }
637
638 }
639
639
640 }
641
642 }
643
644 }
645
646 }
647
648 }
649
649
650 }
651
652 }
653
654 }
655
656 }
657
658 }
659
659
660 }
661
662 }
663
664 }
665
666 }
667
668 }
669
669
670 }
671
672 }
673
674 }
675
676 }
677
678 }
679
679
680 }
681
682 }
683
684 }
685
686 }
687
688 }
689
689
690 }
691
692 }
693
694 }
695
696 }
697
698 }
699
699
700 }
701
702 }
703
704 }
705
706 }
707
708 }
709
709
710 }
711
712 }
713
714 }
715
716 }
717
718 }
719
719
720 }
721
722 }
723
724 }
725
726 }
727
728 }
729
729
730 }
731
732 }
733
734 }
735
736 }
737
738 }
739
739
740 }
741
742 }
743
744 }
745
746 }
747
748 }
749
749
750 }
751
752 }
753
754 }
755
756 }
757
758 }
759
759
760 }
761
762 }
763
764 }
765
766 }
767
768 }
769
769
770 }
771
772 }
773
774 }
775
776 }
777
778 }
779
779
780 }
781
782 }
783
784 }
785
786 }
787
788 }
789
789
790 }
791
792 }
793
794 }
795
796 }
797
798 }
799
799
800 }
801
802 }
803
804 }
805
806 }
807
808 }
809
809
810 }
811
812 }
813
814 }
815
816 }
817
818 }
819
819
820 }
821
822 }
823
824 }
825
826 }
827
828 }
829
829
830 }
831
832 }
833
834 }
835
836 }
837
838 }
839
839
840 }
841
842 }
843
844 }
845
846 }
847
848 }
849
849
850 }
851
852 }
853
854 }
855
856 }
857
858 }
859
859
860 }
861
862 }
863
864 }
865
866 }
867
868 }
869
869
870 }
871
872 }
873
874 }
875
876 }
877
878 }
879
879
880 }
881
882 }
883
884 }
885
886 }
887
888 }
889
889
890 }
891
892 }
893
894 }
895
896 }
897
898 }
899
899
900 }
901
902 }
903
904 }
905
906 }
907
908 }
909
909
910 }
911
912 }
913
914 }
915
916 }
917
918 }
919
919
920 }
921
922 }
923
924 }
925
926 }
927
928 }
929
929
930 }
931
932 }
933
934 }
935
936 }
937
938 }
939
939
940 }
941
942 }
943
944 }
945
946 }
947
948 }
949
949
950 }
951
952 }
953
954 }
955
956 }
957
958 }
959
959
960 }
961
962 }
963
964 }
965
966 }
967
968 }
969
969
970 }
971
972 }
973
974 }
975
976 }
977
978 }
979
979
980 }
981
982 }
983
984 }
985
986 }
987
988 }
989
989
990 }
991
992 }
993
994 }
995
996 }
997
998 }
999
999
1000 }
1001
1002 }
1003
1004 }
1005
1006 }
1007
1008 }
1009
1009
1010 }
1011
1012 }
1013
1014 }
1015
1016 }
1017
1018 }
1019
1019
1020 }
1021
1022 }
1023
1024 }
1025
1026 }
1027
1028 }
1029
1029
1030 }
1031
1032 }
1033
1034 }
1035
1036 }
1037
1038 }
1039
1039
1040 }
1041
1042 }
1043
1044 }
1045
1046 }
1047
1048 }
1049
1049
1050 }
1051
1052 }
1053
1054 }
1055
1056 }
1057
1058 }
1059
1059
1060 }
1061
1062 }
1063
1064 }
1065
1066 }
1067
1068 }
1069
1069
1070 }
1071
1072 }
1073
1074 }
1075
1076 }
1077
1078 }
1079
1079
1080 }
1081
1082 }
1083
1084 }
1085
1086 }
1087
1088 }
1089
1089
1090 }
1091
1092 }
1093
1094 }
1095
1096 }
1097
1098 }
1099
1099
1100 }
1101
1102 }
1103
1104 }
1105
1106 }
1107
1108 }
1109
1109
1110 }
1111
1112 }
1113
1114 }
1115
1116 }
1117
1118 }
1119
1119
1120 }
1121
1122 }
1123
1124 }
1125
1126 }
1127
1128 }
1129
1129
1130 }
1131
1132 }
1133
1134 }
1135
1136 }
1137
1138 }
1139
1139
1140 }
1141
1142 }
1143
1144 }
1145
1146 }
1147
1148 }
1149
1149
1150 }
1151
1152 }
1153
1154 }
1155
1156 }
1157
1158 }
1159
1159
1160 }
1161
1162 }
1163
1164 }
1165
1166 }
1167
1168 }
1169
1169
1170 }
1171
1172 }
1173
1174 }
1175
1176 }
1177
1178 }
1179
1179
1180 }
1181
1182 }
1183
1184 }
1185
1186 }
1187
1188 }
1189
1189
1190 }
1191
1192 }
1193
1194 }
1195
1196 }
1197
1198 }
1199
1199
1200 }
1201
1202 }
1203
1204 }
1205
1206 }
1207
1208 }
1209
1209
1210 }
1211
1212 }
1213
1214 }
1215
1216 }
1217
1218 }
1219
1219
1220 }
1221
1222 }
1223
1224 }
1225
1226 }
1227
1228 }
1229
1229
1230 }
1231
1232 }
1233
1234 }
1235
1236 }
1237
1238 }
1239
1239
1240 }
1241
1242 }
1243
1244 }
1245
1246 }
1247
1248 }
1249
1249
1250 }
1251
1252 }
1253
1254 }
1255
1256 }
1257
1258 }
1259
1259
1260 }
1261
1262 }
1263
1264 }
1265
1266 }
1267
1268 }
1269
1269
1270 }
1271
1272 }
1273
1274 }
1275
1276 }
1277
1278 }
1279
1279
1280 }
1281
1282 }
1283
1284 }
1285
1286 }
1287
1288 }
1289
1289
1290 }
1291
1292 }
1293
1294 }
1295
1296 }
1297
1298 }
1299
1299
1300 }
1301
1302 }
1303
1304 }
1305
1306 }
1307
1308 }
1309
1309
1310 }
1311
1312 }
1313
1314 }
1315
1316 }
1317
1318 }
1319
1319
1320 }
1321
1322 }
1323
1324 }
1325
1326 }
1327
1328 }
1329
1329
1330 }
1331
1332 }
1333
1334 }
1335
1336 }
1337
1338 }
1339
1339
1340 }
1341
1342 }
1343
1344 }
1345
1346 }
1347
1348 }
1349
1349
1350 }
1351
1352 }
1353
1354 }
1355
1356 }
1357
1358 }
1359
1359
1360 }
1361
1362 }
1363
1364 }
1365
1366 }
1367
1368 }
1369
1369
1370 }
1371
1372 }
1373
1374 }
1375
1376 }
1377
1378 }
1379
1379
1380 }
1381
1382 }
1383
1384 }
1385
1386 }
1387
1388 }
1389
1389
1390 }
1391
1392 }
1393
1394 }
1395
1396 }
1397
1398 }
1399
1399
1400 }
1401
1402 }
1403
1404 }
1405
1406 }
1407
1408 }
1409
1409
1410 }
1411
1412 }
1413
1414 }
1415
1416 }
1417
1418 }
1419
1419
1420 }
1421
1422 }
1423
1424 }
1425
1426 }
1427
1428 }
1429
1429
1430 }
1431
1432 }
1433
1434 }
1435
1436 }
1437
1438 }
1439
1439
1440 }
1441
1442 }
1443
1444 }
1445
1446 }
1447
1448 }
1449
1449
1450 }
1451
1452 }
1453
1454 }
1455
1456 }
1457
1458 }
1459
1459
1460 }
1461
1462 }
1463
1464 }
1465
1466 }
1467
1468 }
1469
1469
1470 }
1471
1472 }
1473
1474 }
1475
1476 }
1477
1478 }
1479
1479
1480 }
1481
1482 }
1483
1484 }
1485
1486 }
1487
1488 }
1489
1489
1490 }
1491
1492 }
1493
1494 }
1495
1496 }
1497
1498 }
1499
1499
1500 }
1501
1502 }
1503
1504 }
1505
1506 }
1507
1508 }
1509
1509
1510 }
1511
1512 }
1513
1514 }
1515
1516 }
1517
1518 }
1519
1519
1520 }
1521
1522 }
1523
1524 }
1525
1526 }
1527
1528 }
1529
1529
1530 }
1531
1532 }
1533
1534 }
1535
1536 }
1537
1538 }
1539
1539
1540 }
1541
1542 }
1543
1544 }
1545
1546 }
1547
1548 }
1549
1549
1550 }
1551
1552 }
1553
1554 }
1555
1556 }
1557
1558 }
1559
1559
1560 }
1561
1562 }
1563
1564 }
1565
1566 }
1567
1568 }
1569
1569
1570 }
1571
1572 }
1573
1574 }
1575
1576 }
1577
1578 }
1579
1579
1580 }
1581
1582 }
1583
1584 }
1585
1586 }
1587
1588 }
1589
1589
1590 }
1591
1592 }
1593
1594 }
1595
1596 }
1597
1598 }
1599
1599
1600 }
1601
1602 }
1603
1604 }
1605
1606 }
1607
1608 }
1609
1609
1610 }
1611
1612 }
1613
1614 }
1615
1616 }
1617
1618 }
1619
1619
1620 }
1621
1622 }
1623
1624 }
1625
1626 }
1627
1628 }
1629
1629
1630 }
1631
1632 }
1633
1634 }
1635
1636 }
1637
1638 }
1639
1639
1640 }
1641
1642 }
1643
1644 }
1645
1646 }
1647
1648 }
1649
1649
1650 }
1651
1652 }
1653
1654 }
1655
1656 }
1657
1658 }
1659
1659
1660 }
1661
1662 }
1663
1664 }
1665
1666 }
1667
1668 }
1669
1669
1670 }
1671
1672 }
1673
1674 }
1675
1676 }
1677
1678 }
1679
1679
1680 }
1681
1682 }
1683
1684 }
1685
1686 }
1687
1688 }
1689
1689
1690 }
1691
1692 }
1693
1694 }
1695
1696 }
1697
1698 }
1699
1699
1700 }
1701
1702 }
1703
1704 }
1705
1706 }
1707
1708 }
1709
1709
1710 }
1711
1712 }
1713
1714 }
1715
1716 }
1717
1718 }
1719
1719
1720 }
1721
1722 }
1723
1724 }
1725
1726 }
1727
1728 }
1729
1729
1730 }
1731
1732 }
1733
1734 }
1735
1736 }
1737
1738 }
1739
1739
1740 }
1741
1742 }
1743
1744 }
1745
1746 }
1747
1748 }
1749
1749
1750 }
1751
1752 }
1753
1754 }
1755
1756 }
1757
1758 }
1759
1759
1760 }
1761
1762 }
1763
1764 }
1765
1766 }
1767
1768 }
1769
1769
1770 }
1771
1772 }
1773
1774 }
1775
1776 }
1777
1778 }
1779
1779
1780 }
1781
1782 }
1783
1784 }
1785
1786 }
1787
1788 }
1789
1789
1790 }
1791
1792 }
1793
1794 }
1795
1796 }
1797
1798 }
1799
1799
1800 }
1801
1802 }
1803
1804 }
1805
1806 }
1807
1808 }
1809
1809
1810 }
1811
1812 }
1813
1814 }
1815
1816 }
1817
1818 }
1819
1819
1820 }
1821
1822 }
1823
1824 }
1825
1826 }
1827
1828 }
1829
1829
1830 }
1831
1832 }
1833
1834 }
1835
1836 }
1837
1838 }
1839
1839
1840 }
1841
1842 }
1843
1844 }
1845
1846 }
1847
1848 }
1849
1849
1850 }
1851
1852 }
1853
1854 }
1855
1856 }
1857
1858 }
1859
1859
1860 }
1861
1862 }
1863
1864 }
1865
1866 }
1867
1868 }
1869
1869
1870 }
1871
1872 }
1873
1874 }
1875
1876 }
1877
1878 }
1879
1879
1880 }
1881
1882 }
1883
1884 }
1885
1886 }
1887
1888 }
1889
1889
1890 }
1891
1892 }
1893
1894 }
1895
1896 }
1897
1898 }
1899
1899
1900 }
1901
1902 }
1903
1904 }
1905
1906 }
1907
1908 }
1909
1909
1910 }
1911
1912 }
1913
1914 }
1915
1916 }
1917
1918 }
1919
1919
1920 }
1921
1922 }
1923
1924 }
1925
1926 }
1927
1928 }
1929
1929
1930 }
1931
1932 }
1933
1934 }
1935
1936 }
1937
1938 }
1939
1939
1940 }
1941
1942 }
1943
1944 }
1945
1946 }
1947
1948 }
1949
1949
1950 }
1951
1952 }
1953
1954 }
1955
1956 }
1957
1958 }
1959
1959
1960 }
1961
1962 }
1963
1964 }
1965
1966 }
1967
1968 }
1969
1969
1970 }
1971
1972 }
1973
1974 }
1975
1976 }
1977
1978 }
1979
1979
1980 }
1981
1982 }
1983
1984 }
1985
1986 }
1987
1988 }
1989
1989
1990 }
1991
1992 }
1993
1994 }
1995
1996 }
1997
1998 }
1999
1999
2000 }
2001
2002 }
2003
2004 }
2005
2006 }
2007
2008 }
2009
2009
2010 }
2011
2012 }
2013
2014 }
2015
2016 }
2017
2018 }
2019
2019
2020 }
2021
2022 }
2023
2024 }
2025
2026 }
2027
2028 }
2029
2029
2030 }
2031
2032 }
2033
2034 }
2035
2036 }
2037
2038 }
2039
2039
2040 }
2041
2042 }
2043
2044 }
2045
2046 }
2047
2048 }
2049
2049
2050 }
2051
2052 }
2053
2054 }
2055
2056 }
2057
2058 }
2059
2059
2060 }
2061
2062 }
2063
2064 }
2065
2066 }
2067
2068 }
2069
2069
2070 }
2071
2072 }
2073
2074 }
2075
2076 }
2077
2078 }
2079
2079
2080 }
2081
2082 }
2083
2084 }
2085
2086 }
2087
2088 }
2089
2089
2090 }
2091
2092 }
2093
2094 }
2095
2096 }
2097
2098 }
2099
2099
2100 }
2101
2102 }
2103
2104 }
2105
2106 }
2107
2108 }
2109
2109
2110 }
2111
2112 }
2113
2114 }
2115
2116 }
2117
2118 }
2119
2119
2120 }
2121
2122 }
2123
2124 }
2125
2126 }
2127
2128 }
2129
2129
2130 }
2131
2132 }
2133
2134 }
2135
2136 }
2137
2138 }
2139
2139
2140 }
2141
2142 }
2143
2144 }
2145
2146 }
2147
2148 }
2149
2149
2150 }
2151
2152 }
2153
2154 }
2155
2156 }
2157
2158 }
2159
2159
2160 }
2161
2162 }
2163
2164 }
2165
2166 }
2167
2168 }
2169
2169
2170 }
2171
2172 }
2173
2174 }
2175
2176 }
2177
2178 }
2179
2179
2180 }
2181
2182 }
2183
2184 }
2185
2186 }
2187
2188 }
2189
2189
2190 }
2191
2192 }
2193
2194 }
2195
2196 }
2197
2198 }
2199
2199
2200 }
2201
2202 }
2203
2204 }
2205
2206 }
2207
2208 }
2209
2209
2210 }
2211
2212 }
2213
2214 }
2215
2216 }
2217
2218 }
2219
2219
2220 }
2221
2222 }
2223
2224 }
2225
2226 }
2227
2228 }
2229
2229
2230 }
2231
2232 }
2233
2234 }
2235
2236 }
2237
2238 }
2239
2239
2240 }
2241
2242 }

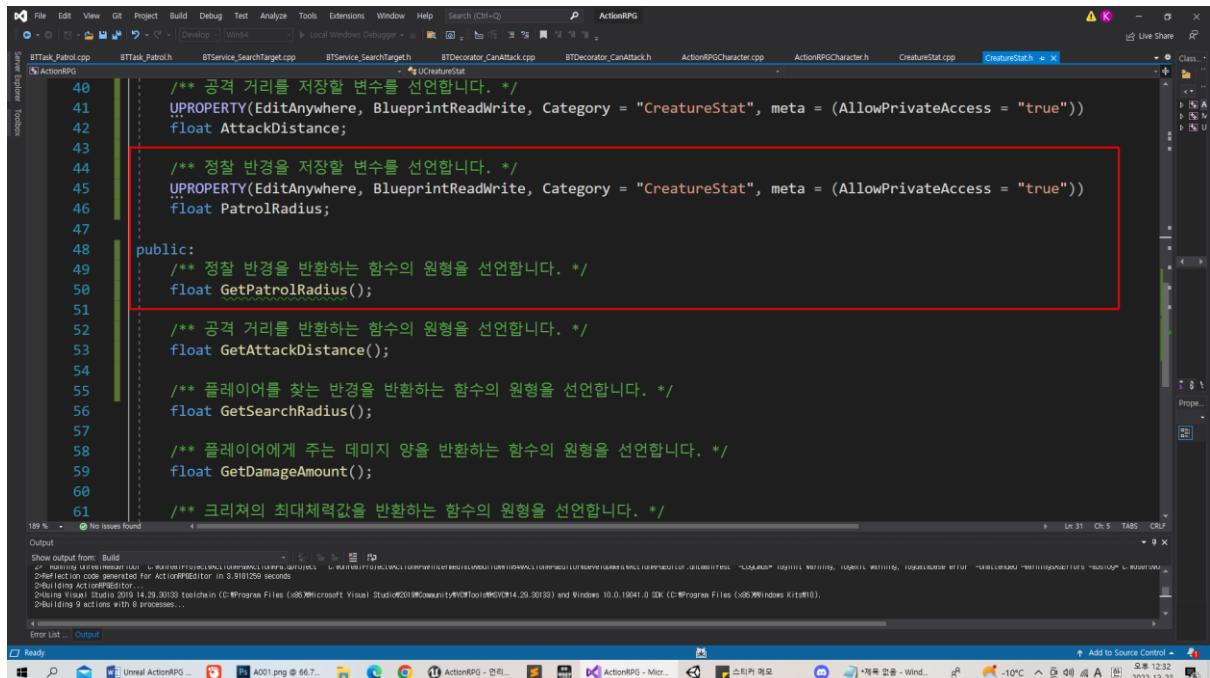
```



```

ActionRPG
File Edit View GB Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+F) ActionRPG
BTTTask_Patrol.cpp BTTTask_Patrol.h BTService_SearchTarget.cpp BTService_SearchTarget.h BTDecorator_CanAttack.cpp BTDecorator_CanAttack.h ActionRPGCharacter.cpp ActionRPGCharacter.h CreatureStat.cpp CreatureStat.h
Live Share
ActionRPG
ActionRPG
28  /** 크리쳐의 AIController를 가져 옵니다. */
29  ACreatureAIController* CreatureAIController = Cast<ACreatureAIController>(OwnerComp.GetAIOwner());
30
31  if (NavigationSystem && CreatureAIController && CreatureAIController->GetPawn())
32  {
33      /**
34      패트를 할 반경을 설정합니다.
35      언리얼에서 단위는 센티미터입니다. 반경을 6미터로 설정합니다.
36      */
37      const float PatrolRadius = 600.0f;
38      /** 임시 패트를 위치를 저장할 변수를 선언합니다. */
39      FNavigation PatrolLocation;
40      /** 위치를 찾았는지 확인할 블리언 변수를 선언해 줍니다. */
41      const bool bIsFound =
42          NavigationSystem->GetRandomPointInNavigableRadius(
43              CreatureAIController->GetPawn()->GetActorLocation(), PatrolRadius, PatrolLocation);
44      /** 만일 패트를 할 랜덤 위치를 찾았다면? */
45      if (bIsFound)
46      {
47          /**
48          패트를 할 랜덤위치를 찾았다고 블랙 보드에 저장합니다.
49          */
50      }
51  }
52
53  void AActionRPGCharacter::Patrol()
54  {
55      if (CreatureAIController)
56      {
57          if (NavigationSystem)
58          {
59              BTTTask_Patrol* Task = new BTTTask_Patrol();
60              Task->Init(CreatureAIController, PatrolRadius);
61              Task->Execute();
62          }
63      }
64  }
65
66  void AActionRPGCharacter::SearchTarget()
67  {
68      if (CreatureAIController)
69      {
70          if (NavigationSystem)
71          {
72              BTService_SearchTarget* Service = new BTService_SearchTarget();
73              Service->Init(CreatureAIController);
74              Service->Execute();
75          }
76      }
77  }
78
79  void AActionRPGCharacter::Attack()
80  {
81      if (CreatureAIController)
82      {
83          if (NavigationSystem)
84          {
85              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
86              Decorator->Init(CreatureAIController);
87              Decorator->Execute();
88          }
89      }
90  }
91
92  void AActionRPGCharacter::Decorate()
93  {
94      if (CreatureAIController)
95      {
96          if (NavigationSystem)
97          {
98              BTService_SearchTarget* Service = new BTService_SearchTarget();
99              Service->Init(CreatureAIController);
100             Service->Execute();
101         }
102     }
103  }
104
105  void AActionRPGCharacter::HandleSearchTarget()
106  {
107      if (CreatureAIController)
108      {
109          if (NavigationSystem)
110          {
111              BTService_SearchTarget* Service = new BTService_SearchTarget();
112              Service->Init(CreatureAIController);
113              Service->Execute();
114          }
115      }
116  }
117
118  void AActionRPGCharacter::HandleAttack()
119  {
120      if (CreatureAIController)
121      {
122          if (NavigationSystem)
123          {
124              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
125              Decorator->Init(CreatureAIController);
126              Decorator->Execute();
127          }
128      }
129  }
130
131  void AActionRPGCharacter::HandleDecorate()
132  {
133      if (CreatureAIController)
134      {
135          if (NavigationSystem)
136          {
137              BTService_SearchTarget* Service = new BTService_SearchTarget();
138              Service->Init(CreatureAIController);
139              Service->Execute();
140          }
141      }
142  }
143
144  void AActionRPGCharacter::HandlePatrol()
145  {
146      if (CreatureAIController)
147      {
148          if (NavigationSystem)
149          {
150              BTTTask_Patrol* Task = new BTTTask_Patrol();
151              Task->Init(CreatureAIController, PatrolRadius);
152              Task->Execute();
153          }
154      }
155  }
156
157  void AActionRPGCharacter::HandleSearch()
158  {
159      if (CreatureAIController)
160      {
161          if (NavigationSystem)
162          {
163              BTService_SearchTarget* Service = new BTService_SearchTarget();
164              Service->Init(CreatureAIController);
165              Service->Execute();
166          }
167      }
168  }
169
170  void AActionRPGCharacter::HandleAttack()
171  {
172      if (CreatureAIController)
173      {
174          if (NavigationSystem)
175          {
176              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
177              Decorator->Init(CreatureAIController);
178              Decorator->Execute();
179          }
180      }
181  }
182
183  void AActionRPGCharacter::HandleDecorate()
184  {
185      if (CreatureAIController)
186      {
187          if (NavigationSystem)
188          {
189              BTService_SearchTarget* Service = new BTService_SearchTarget();
190              Service->Init(CreatureAIController);
191              Service->Execute();
192          }
193      }
194  }
195
196  void AActionRPGCharacter::HandlePatrol()
197  {
198      if (CreatureAIController)
199      {
200          if (NavigationSystem)
201          {
202              BTTTask_Patrol* Task = new BTTTask_Patrol();
203              Task->Init(CreatureAIController, PatrolRadius);
204              Task->Execute();
205          }
206      }
207  }
208
209  void AActionRPGCharacter::HandleSearch()
210  {
211      if (CreatureAIController)
212      {
213          if (NavigationSystem)
214          {
215              BTService_SearchTarget* Service = new BTService_SearchTarget();
216              Service->Init(CreatureAIController);
217              Service->Execute();
218          }
219      }
220  }
221
222  void AActionRPGCharacter::HandleAttack()
223  {
224      if (CreatureAIController)
225      {
226          if (NavigationSystem)
227          {
228              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
229              Decorator->Init(CreatureAIController);
230              Decorator->Execute();
231          }
232      }
233  }
234
235  void AActionRPGCharacter::HandleDecorate()
236  {
237      if (CreatureAIController)
238      {
239          if (NavigationSystem)
240          {
241              BTService_SearchTarget* Service = new BTService_SearchTarget();
242              Service->Init(CreatureAIController);
243              Service->Execute();
244          }
245      }
246  }
247
248  void AActionRPGCharacter::HandlePatrol()
249  {
250      if (CreatureAIController)
251      {
252          if (NavigationSystem)
253          {
254              BTTTask_Patrol* Task = new BTTTask_Patrol();
255              Task->Init(CreatureAIController, PatrolRadius);
256              Task->Execute();
257          }
258      }
259  }
260
261  void AActionRPGCharacter::HandleSearch()
262  {
263      if (CreatureAIController)
264      {
265          if (NavigationSystem)
266          {
267              BTService_SearchTarget* Service = new BTService_SearchTarget();
268              Service->Init(CreatureAIController);
269              Service->Execute();
270          }
271      }
272  }
273
274  void AActionRPGCharacter::HandleAttack()
275  {
276      if (CreatureAIController)
277      {
278          if (NavigationSystem)
279          {
280              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
281              Decorator->Init(CreatureAIController);
282              Decorator->Execute();
283          }
284      }
285  }
286
287  void AActionRPGCharacter::HandleDecorate()
288  {
289      if (CreatureAIController)
290      {
291          if (NavigationSystem)
292          {
293              BTService_SearchTarget* Service = new BTService_SearchTarget();
294              Service->Init(CreatureAIController);
295              Service->Execute();
296          }
297      }
298  }
299
300  void AActionRPGCharacter::HandlePatrol()
301  {
302      if (CreatureAIController)
303      {
304          if (NavigationSystem)
305          {
306              BTTTask_Patrol* Task = new BTTTask_Patrol();
307              Task->Init(CreatureAIController, PatrolRadius);
308              Task->Execute();
309          }
310      }
311  }
312
313  void AActionRPGCharacter::HandleSearch()
314  {
315      if (CreatureAIController)
316      {
317          if (NavigationSystem)
318          {
319              BTService_SearchTarget* Service = new BTService_SearchTarget();
320              Service->Init(CreatureAIController);
321              Service->Execute();
322          }
323      }
324  }
325
326  void AActionRPGCharacter::HandleAttack()
327  {
328      if (CreatureAIController)
329      {
330          if (NavigationSystem)
331          {
332              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
333              Decorator->Init(CreatureAIController);
334              Decorator->Execute();
335          }
336      }
337  }
338
339  void AActionRPGCharacter::HandleDecorate()
340  {
341      if (CreatureAIController)
342      {
343          if (NavigationSystem)
344          {
345              BTService_SearchTarget* Service = new BTService_SearchTarget();
346              Service->Init(CreatureAIController);
347              Service->Execute();
348          }
349      }
350  }
351
352  void AActionRPGCharacter::HandlePatrol()
353  {
354      if (CreatureAIController)
355      {
356          if (NavigationSystem)
357          {
358              BTTTask_Patrol* Task = new BTTTask_Patrol();
359              Task->Init(CreatureAIController, PatrolRadius);
360              Task->Execute();
361          }
362      }
363  }
364
365  void AActionRPGCharacter::HandleSearch()
366  {
367      if (CreatureAIController)
368      {
369          if (NavigationSystem)
370          {
371              BTService_SearchTarget* Service = new BTService_SearchTarget();
372              Service->Init(CreatureAIController);
373              Service->Execute();
374          }
375      }
376  }
377
378  void AActionRPGCharacter::HandleAttack()
379  {
380      if (CreatureAIController)
381      {
382          if (NavigationSystem)
383          {
384              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
385              Decorator->Init(CreatureAIController);
386              Decorator->Execute();
387          }
388      }
389  }
390
391  void AActionRPGCharacter::HandleDecorate()
392  {
393      if (CreatureAIController)
394      {
395          if (NavigationSystem)
396          {
397              BTService_SearchTarget* Service = new BTService_SearchTarget();
398              Service->Init(CreatureAIController);
399              Service->Execute();
400          }
401      }
402  }
403
404  void AActionRPGCharacter::HandlePatrol()
405  {
406      if (CreatureAIController)
407      {
408          if (NavigationSystem)
409          {
410              BTTTask_Patrol* Task = new BTTTask_Patrol();
411              Task->Init(CreatureAIController, PatrolRadius);
412              Task->Execute();
413          }
414      }
415  }
416
417  void AActionRPGCharacter::HandleSearch()
418  {
419      if (CreatureAIController)
420      {
421          if (NavigationSystem)
422          {
423              BTService_SearchTarget* Service = new BTService_SearchTarget();
424              Service->Init(CreatureAIController);
425              Service->Execute();
426          }
427      }
428  }
429
430  void AActionRPGCharacter::HandleAttack()
431  {
432      if (CreatureAIController)
433      {
434          if (NavigationSystem)
435          {
436              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
437              Decorator->Init(CreatureAIController);
438              Decorator->Execute();
439          }
440      }
441  }
442
443  void AActionRPGCharacter::HandleDecorate()
444  {
445      if (CreatureAIController)
446      {
447          if (NavigationSystem)
448          {
449              BTService_SearchTarget* Service = new BTService_SearchTarget();
450              Service->Init(CreatureAIController);
451              Service->Execute();
452          }
453      }
454  }
455
456  void AActionRPGCharacter::HandlePatrol()
457  {
458      if (CreatureAIController)
459      {
460          if (NavigationSystem)
461          {
462              BTTTask_Patrol* Task = new BTTTask_Patrol();
463              Task->Init(CreatureAIController, PatrolRadius);
464              Task->Execute();
465          }
466      }
467  }
468
469  void AActionRPGCharacter::HandleSearch()
470  {
471      if (CreatureAIController)
472      {
473          if (NavigationSystem)
474          {
475              BTService_SearchTarget* Service = new BTService_SearchTarget();
476              Service->Init(CreatureAIController);
477              Service->Execute();
478          }
479      }
480  }
481
482  void AActionRPGCharacter::HandleAttack()
483  {
484      if (CreatureAIController)
485      {
486          if (NavigationSystem)
487          {
488              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
489              Decorator->Init(CreatureAIController);
490              Decorator->Execute();
491          }
492      }
493  }
494
495  void AActionRPGCharacter::HandleDecorate()
496  {
497      if (CreatureAIController)
498      {
499          if (NavigationSystem)
500          {
501              BTService_SearchTarget* Service = new BTService_SearchTarget();
502              Service->Init(CreatureAIController);
503              Service->Execute();
504          }
505      }
506  }
507
508  void AActionRPGCharacter::HandlePatrol()
509  {
510      if (CreatureAIController)
511      {
512          if (NavigationSystem)
513          {
514              BTTTask_Patrol* Task = new BTTTask_Patrol();
515              Task->Init(CreatureAIController, PatrolRadius);
516              Task->Execute();
517          }
518      }
519  }
520
521  void AActionRPGCharacter::HandleSearch()
522  {
523      if (CreatureAIController)
524      {
525          if (NavigationSystem)
526          {
527              BTService_SearchTarget* Service = new BTService_SearchTarget();
528              Service->Init(CreatureAIController);
529              Service->Execute();
530          }
531      }
532  }
533
534  void AActionRPGCharacter::HandleAttack()
535  {
536      if (CreatureAIController)
537      {
538          if (NavigationSystem)
539          {
540              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
541              Decorator->Init(CreatureAIController);
542              Decorator->Execute();
543          }
544      }
545  }
546
547  void AActionRPGCharacter::HandleDecorate()
548  {
549      if (CreatureAIController)
550      {
551          if (NavigationSystem)
552          {
553              BTService_SearchTarget* Service = new BTService_SearchTarget();
554              Service->Init(CreatureAIController);
555              Service->Execute();
556          }
557      }
558  }
559
560  void AActionRPGCharacter::HandlePatrol()
561  {
562      if (CreatureAIController)
563      {
564          if (NavigationSystem)
565          {
566              BTTTask_Patrol* Task = new BTTTask_Patrol();
567              Task->Init(CreatureAIController, PatrolRadius);
568              Task->Execute();
569          }
570      }
571  }
572
573  void AActionRPGCharacter::HandleSearch()
574  {
575      if (CreatureAIController)
576      {
577          if (NavigationSystem)
578          {
579              BTService_SearchTarget* Service = new BTService_SearchTarget();
580              Service->Init(CreatureAIController);
581              Service->Execute();
582          }
583      }
584  }
585
586  void AActionRPGCharacter::HandleAttack()
587  {
588      if (CreatureAIController)
589      {
590          if (NavigationSystem)
591          {
592              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
593              Decorator->Init(CreatureAIController);
594              Decorator->Execute();
595          }
596      }
597  }
598
599  void AActionRPGCharacter::HandleDecorate()
600  {
601      if (CreatureAIController)
602      {
603          if (NavigationSystem)
604          {
605              BTService_SearchTarget* Service = new BTService_SearchTarget();
606              Service->Init(CreatureAIController);
607              Service->Execute();
608          }
609      }
610  }
611
612  void AActionRPGCharacter::HandlePatrol()
613  {
614      if (CreatureAIController)
615      {
616          if (NavigationSystem)
617          {
618              BTTTask_Patrol* Task = new BTTTask_Patrol();
619              Task->Init(CreatureAIController, PatrolRadius);
620              Task->Execute();
621          }
622      }
623  }
624
625  void AActionRPGCharacter::HandleSearch()
626  {
627      if (CreatureAIController)
628      {
629          if (NavigationSystem)
630          {
631              BTService_SearchTarget* Service = new BTService_SearchTarget();
632              Service->Init(CreatureAIController);
633              Service->Execute();
634          }
635      }
636  }
637
638  void AActionRPGCharacter::HandleAttack()
639  {
640      if (CreatureAIController)
641      {
642          if (NavigationSystem)
643          {
644              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
645              Decorator->Init(CreatureAIController);
646              Decorator->Execute();
647          }
648      }
649  }
650
651  void AActionRPGCharacter::HandleDecorate()
652  {
653      if (CreatureAIController)
654      {
655          if (NavigationSystem)
656          {
657              BTService_SearchTarget* Service = new BTService_SearchTarget();
658              Service->Init(CreatureAIController);
659              Service->Execute();
660          }
661      }
662  }
663
664  void AActionRPGCharacter::HandlePatrol()
665  {
666      if (CreatureAIController)
667      {
668          if (NavigationSystem)
669          {
670              BTTTask_Patrol* Task = new BTTTask_Patrol();
671              Task->Init(CreatureAIController, PatrolRadius);
672              Task->Execute();
673          }
674      }
675  }
676
677  void AActionRPGCharacter::HandleSearch()
678  {
679      if (CreatureAIController)
680      {
681          if (NavigationSystem)
682          {
683              BTService_SearchTarget* Service = new BTService_SearchTarget();
684              Service->Init(CreatureAIController);
685              Service->Execute();
686          }
687      }
688  }
689
690  void AActionRPGCharacter::HandleAttack()
691  {
692      if (CreatureAIController)
693      {
694          if (NavigationSystem)
695          {
696              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
697              Decorator->Init(CreatureAIController);
698              Decorator->Execute();
699          }
700      }
701  }
702
703  void AActionRPGCharacter::HandleDecorate()
704  {
705      if (CreatureAIController)
706      {
707          if (NavigationSystem)
708          {
709              BTService_SearchTarget* Service = new BTService_SearchTarget();
710              Service->Init(CreatureAIController);
711              Service->Execute();
712          }
713      }
714  }
715
716  void AActionRPGCharacter::HandlePatrol()
717  {
718      if (CreatureAIController)
719      {
720          if (NavigationSystem)
721          {
722              BTTTask_Patrol* Task = new BTTTask_Patrol();
723              Task->Init(CreatureAIController, PatrolRadius);
724              Task->Execute();
725          }
726      }
727  }
728
729  void AActionRPGCharacter::HandleSearch()
730  {
731      if (CreatureAIController)
732      {
733          if (NavigationSystem)
734          {
735              BTService_SearchTarget* Service = new BTService_SearchTarget();
736              Service->Init(CreatureAIController);
737              Service->Execute();
738          }
739      }
740  }
741
742  void AActionRPGCharacter::HandleAttack()
743  {
744      if (CreatureAIController)
745      {
746          if (NavigationSystem)
747          {
748              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
749              Decorator->Init(CreatureAIController);
750              Decorator->Execute();
751          }
752      }
753  }
754
755  void AActionRPGCharacter::HandleDecorate()
756  {
757      if (CreatureAIController)
758      {
759          if (NavigationSystem)
760          {
761              BTService_SearchTarget* Service = new BTService_SearchTarget();
762              Service->Init(CreatureAIController);
763              Service->Execute();
764          }
765      }
766  }
767
768  void AActionRPGCharacter::HandlePatrol()
769  {
770      if (CreatureAIController)
771      {
772          if (NavigationSystem)
773          {
774              BTTTask_Patrol* Task = new BTTTask_Patrol();
775              Task->Init(CreatureAIController, PatrolRadius);
776              Task->Execute();
777          }
778      }
779  }
780
781  void AActionRPGCharacter::HandleSearch()
782  {
783      if (CreatureAIController)
784      {
785          if (NavigationSystem)
786          {
787              BTService_SearchTarget* Service = new BTService_SearchTarget();
788              Service->Init(CreatureAIController);
789              Service->Execute();
790          }
791      }
792  }
793
794  void AActionRPGCharacter::HandleAttack()
795  {
796      if (CreatureAIController)
797      {
798          if (NavigationSystem)
799          {
800              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
801              Decorator->Init(CreatureAIController);
802              Decorator->Execute();
803          }
804      }
805  }
806
807  void AActionRPGCharacter::HandleDecorate()
808  {
809      if (CreatureAIController)
810      {
811          if (NavigationSystem)
812          {
813              BTService_SearchTarget* Service = new BTService_SearchTarget();
814              Service->Init(CreatureAIController);
815              Service->Execute();
816          }
817      }
818  }
819
820  void AActionRPGCharacter::HandlePatrol()
821  {
822      if (CreatureAIController)
823      {
824          if (NavigationSystem)
825          {
826              BTTTask_Patrol* Task = new BTTTask_Patrol();
827              Task->Init(CreatureAIController, PatrolRadius);
828              Task->Execute();
829          }
830      }
831  }
832
833  void AActionRPGCharacter::HandleSearch()
834  {
835      if (CreatureAIController)
836      {
837          if (NavigationSystem)
838          {
839              BTService_SearchTarget* Service = new BTService_SearchTarget();
840              Service->Init(CreatureAIController);
841              Service->Execute();
842          }
843      }
844  }
845
846  void AActionRPGCharacter::HandleAttack()
847  {
848      if (CreatureAIController)
849      {
850          if (NavigationSystem)
851          {
852              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
853              Decorator->Init(CreatureAIController);
854              Decorator->Execute();
855          }
856      }
857  }
858
859  void AActionRPGCharacter::HandleDecorate()
860  {
861      if (CreatureAIController)
862      {
863          if (NavigationSystem)
864          {
865              BTService_SearchTarget* Service = new BTService_SearchTarget();
866              Service->Init(CreatureAIController);
867              Service->Execute();
868          }
869      }
870  }
871
872  void AActionRPGCharacter::HandlePatrol()
873  {
874      if (CreatureAIController)
875      {
876          if (NavigationSystem)
877          {
878              BTTTask_Patrol* Task = new BTTTask_Patrol();
879              Task->Init(CreatureAIController, PatrolRadius);
880              Task->Execute();
881          }
882      }
883  }
884
885  void AActionRPGCharacter::HandleSearch()
886  {
887      if (CreatureAIController)
888      {
889          if (NavigationSystem)
890          {
891              BTService_SearchTarget* Service = new BTService_SearchTarget();
892              Service->Init(CreatureAIController);
893              Service->Execute();
894          }
895      }
896  }
897
898  void AActionRPGCharacter::HandleAttack()
899  {
900      if (CreatureAIController)
901      {
902          if (NavigationSystem)
903          {
904              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
905              Decorator->Init(CreatureAIController);
906              Decorator->Execute();
907          }
908      }
909  }
910
911  void AActionRPGCharacter::HandleDecorate()
912  {
913      if (CreatureAIController)
914      {
915          if (NavigationSystem)
916          {
917              BTService_SearchTarget* Service = new BTService_SearchTarget();
918              Service->Init(CreatureAIController);
919              Service->Execute();
920          }
921      }
922  }
923
924  void AActionRPGCharacter::HandlePatrol()
925  {
926      if (CreatureAIController)
927      {
928          if (NavigationSystem)
929          {
930              BTTTask_Patrol* Task = new BTTTask_Patrol();
931              Task->Init(CreatureAIController, PatrolRadius);
932              Task->Execute();
933          }
934      }
935  }
936
937  void AActionRPGCharacter::HandleSearch()
938  {
939      if (CreatureAIController)
940      {
941          if (NavigationSystem)
942          {
943              BTService_SearchTarget* Service = new BTService_SearchTarget();
944              Service->Init(CreatureAIController);
945              Service->Execute();
946          }
947      }
948  }
949
950  void AActionRPGCharacter::HandleAttack()
951  {
952      if (CreatureAIController)
953      {
954          if (NavigationSystem)
955          {
956              BTDecorator_CanAttack* Decorator = new BTDecorator_CanAttack();
957              Decorator->Init(CreatureAIController);
958              Decorator->Execute();
959          }
960      }
961  }
962
963  void AActionRPGCharacter::HandleDecorate()
964  {
965      if (CreatureAIController)
966      {
967          if (NavigationSystem)
968          {
969              BTService_SearchTarget* Service = new BTService_SearchTarget();
970              Service->Init(CreatureAIController);
971              Service->Execute();
972          }
973      }
974  }
975
976  void AActionRPGCharacter::HandlePatrol()
977  {
978      if (CreatureAIController)
979      {
980          if (NavigationSystem)
981          {
982              BTTTask_Patrol* Task = new BTTTask_Patrol();
983              Task->Init(CreatureAIController, PatrolRadius);
984              Task->Execute();
985          }
986      }
987  }
988
989  void AActionRPGCharacter::HandleSearch()
990  {
991      if (CreatureAIController)
992      {
993          if (NavigationSystem)
994          {
995              BTService_SearchTarget* Service = new BTService_SearchTarget();
996              Service->Init(CreatureAIController);
997              Service->Execute();
998          }
999      }
1000 }

```

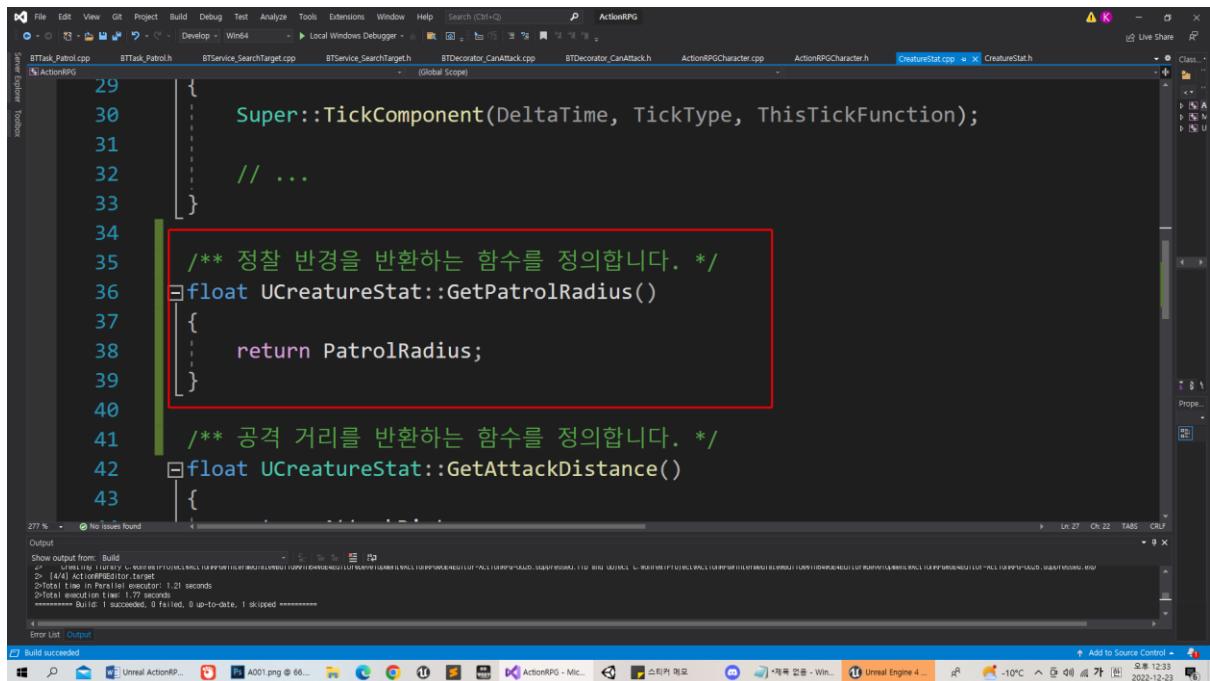


```
40  /** 공격 거리를 저장할 변수를 선언합니다. */
41  UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = "true"))
42  float AttackDistance;
43
44  /** 정찰 반경을 저장할 변수를 선언합니다. */
45  UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (AllowPrivateAccess = "true"))
46  float PatrolRadius;
47
48  public:
49  /** 정찰 반경을 반환하는 함수의 원형을 선언합니다. */
50  float GetPatrolRadius();
51
52  /** 공격 거리를 반환하는 함수의 원형을 선언합니다. */
53  float GetAttackDistance();
54
55  /** 플레이어를 찾는 반경을 반환하는 함수의 원형을 선언합니다. */
56  float GetSearchRadius();
57
58  /** 플레이어에게 주는 데미지 양을 반환하는 함수의 원형을 선언합니다. */
59  float GetDamageAmount();
60
61  /** 크리쳐의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
```

Output
Build succeeded - 1/14(1) ActionRPGEditor Target
2/Total time in Parallel executor: 1.21 seconds
2/Total execution time: 1.77 seconds
***** Build 1 succeeded, 0 failed, 0 up-to-date, 1 skipped *****

Error List Output

구현해 줍니다.

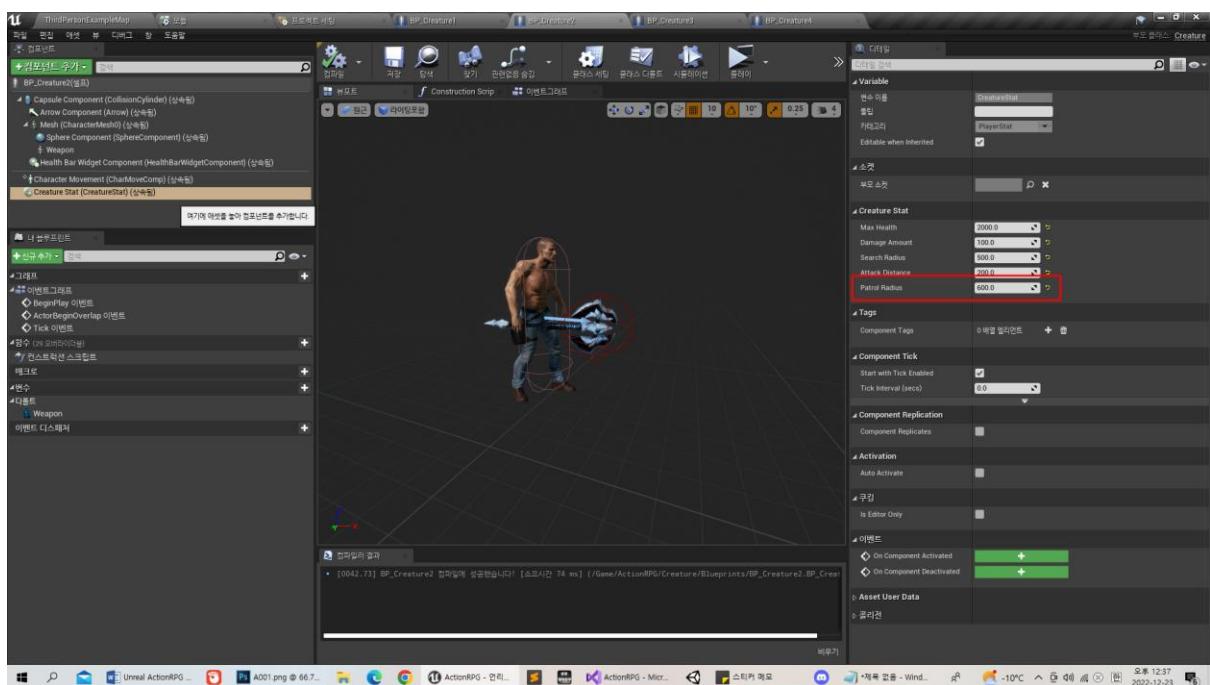
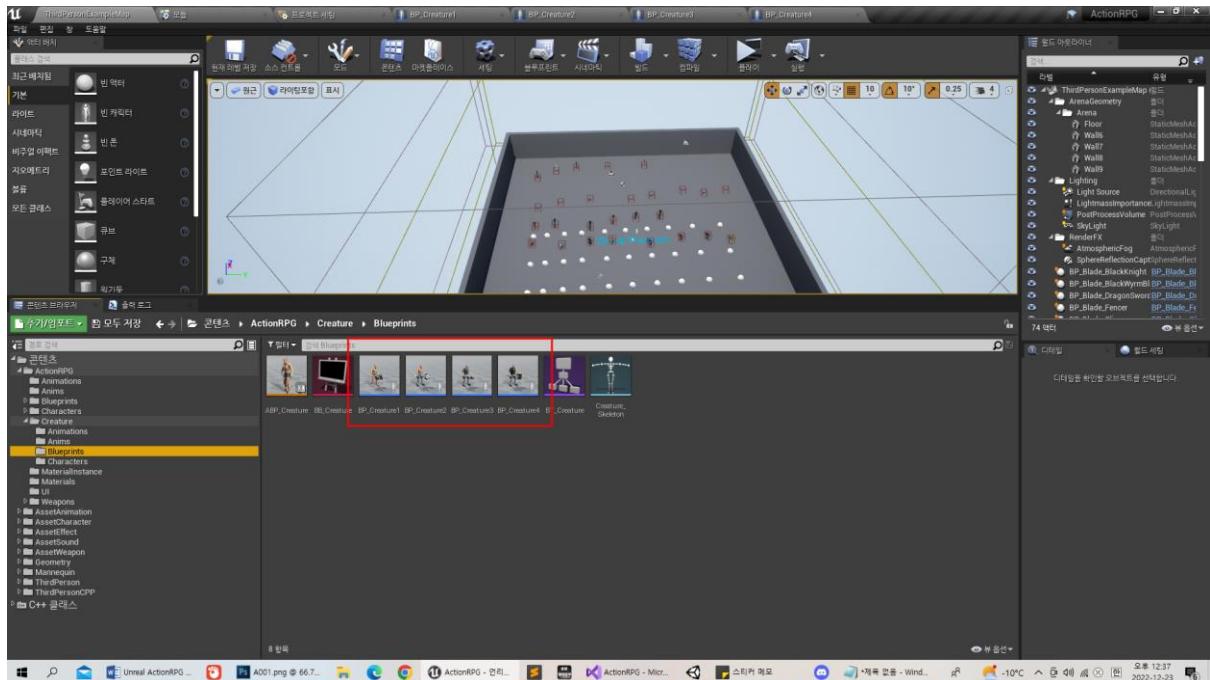


```
29  {
30     Super::TickComponent(DeltaTime, TickType, ThisTickFunction);
31
32     // ...
33 }
34
35  /** 정찰 반경을 반환하는 함수를 정의합니다. */
36  float UCreatureStat::GetPatrolRadius()
37  {
38     return PatrolRadius;
39 }
40
41  /** 공격 거리를 반환하는 함수를 정의합니다. */
42  float UCreatureStat::GetAttackDistance()
43  {
```

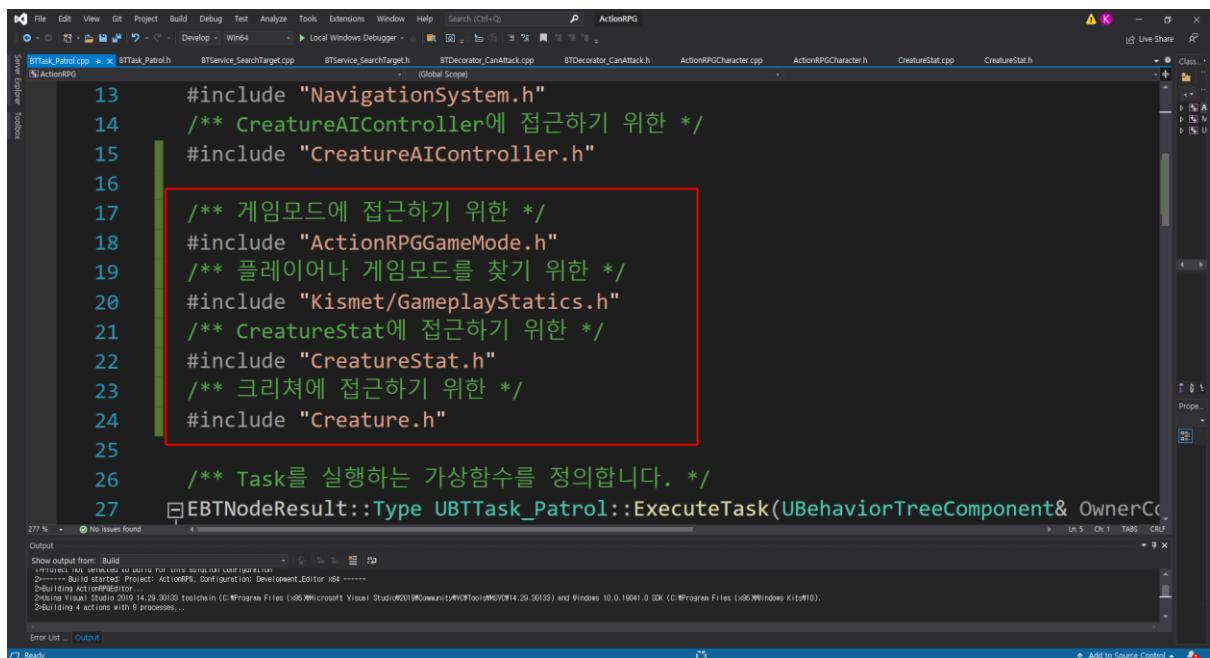
Output
Build succeeded - 1/14(1) ActionRPGEditor Target
2/Total time in Parallel executor: 1.21 seconds
2/Total execution time: 1.77 seconds
***** Build 1 succeeded, 0 failed, 0 up-to-date, 1 skipped *****

Error List Output

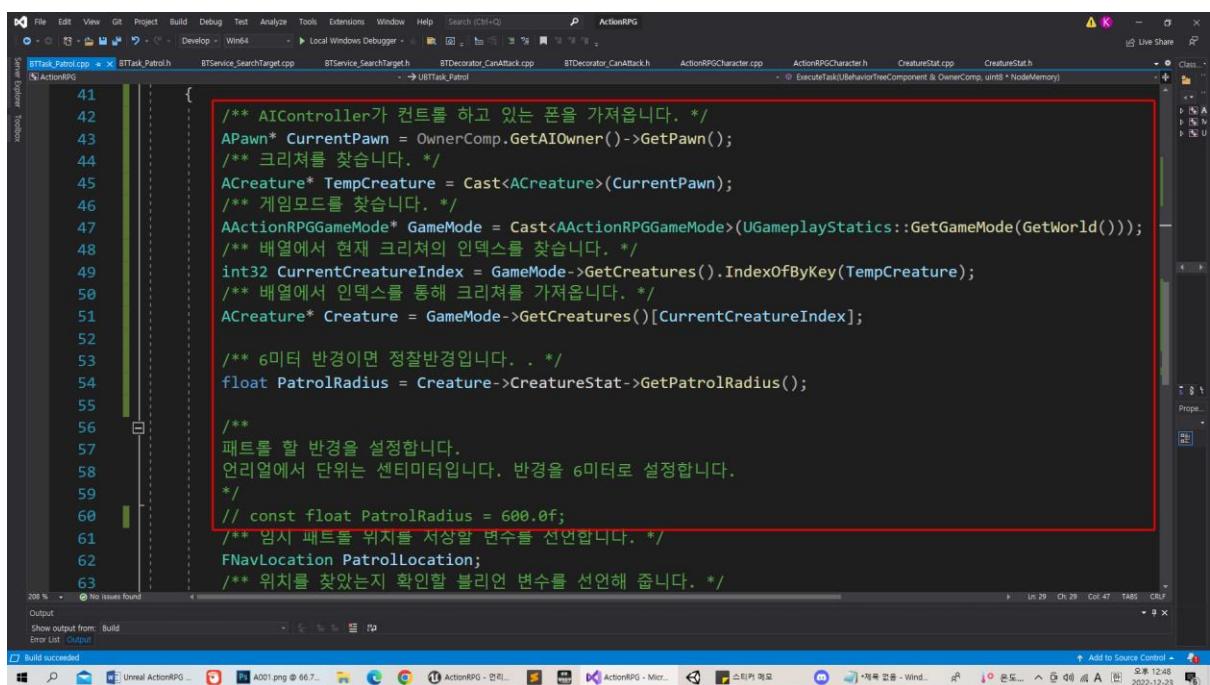
크리쳐 블루프린트에서 적용해 주도록 합니다.



BTDecorator_CanAttack 클래스에서 수정해 줍니다.



```
13 #include "NavigationSystem.h"
14 /** CreatureAIController에 접근하기 위한 */
15 #include "CreatureAIController.h"
16
17 /** 게임모드에 접근하기 위한 */
18 #include "ActionRPGGameMode.h"
19 /** 플레이어나 게임모드를 찾기 위한 */
20 #include "Kismet/GameplayStatics.h"
21 /** CreatureStat에 접근하기 위한 */
22 #include "CreatureStat.h"
23 /** 크리쳐에 접근하기 위한 */
24 #include "Creature.h"
25
26 /** Task를 실행하는 가상함수를 정의합니다. */
27 EBTNodeResult::Type UBTTask_Patrol::ExecuteTask(UBehaviorTreeComponent& OwnerCo...
```



```
41 {
42     /** AIController가 컨트롤 하고 있는 폰을 가져옵니다. */
43     APawn* CurrentPawn = OwnerComp.GetAIOWner()->GetPawn();
44     /** 크리쳐를 찾습니다. */
45     ACreature* TempCreature = Cast<ACreature>(CurrentPawn);
46     /** 게임모드를 찾습니다. */
47     AActionRPGGameMode* GameMode = Cast<AActionRPGGameMode>(UGameplayStatics::GetGameMode(GetWorld()));
48     /** 배열에서 현재 크리쳐의 인덱스를 찾습니다. */
49     int32 CurrentCreatureIndex = GameMode->GetCreatures().IndexByKey(TempCreature);
50     /** 배열에서 인덱스를 통해 크리쳐를 가져옵니다. */
51     ACreature* Creature = GameMode->GetCreatures()[CurrentCreatureIndex];
52
53     /** 6미터 반경이면 정찰반경입니다. . */
54     float PatrolRadius = Creature->CreatureStat->GetPatrolRadius();
55
56     /**
57     패트를 할 반경을 설정합니다.
58     언리얼에서 단위는 센티미터입니다. 반경을 6미터로 설정합니다.
59     */
60     // const float PatrolRadius = 600.0f;
61     /** 임시 패트를 위치를 설정할 변수를 선언합니다. */
62     FNavLocation PatrolLocation;
63     /** 위치를 찾았는지 확인할 불리언 변수를 선언해 줍니다. */
```