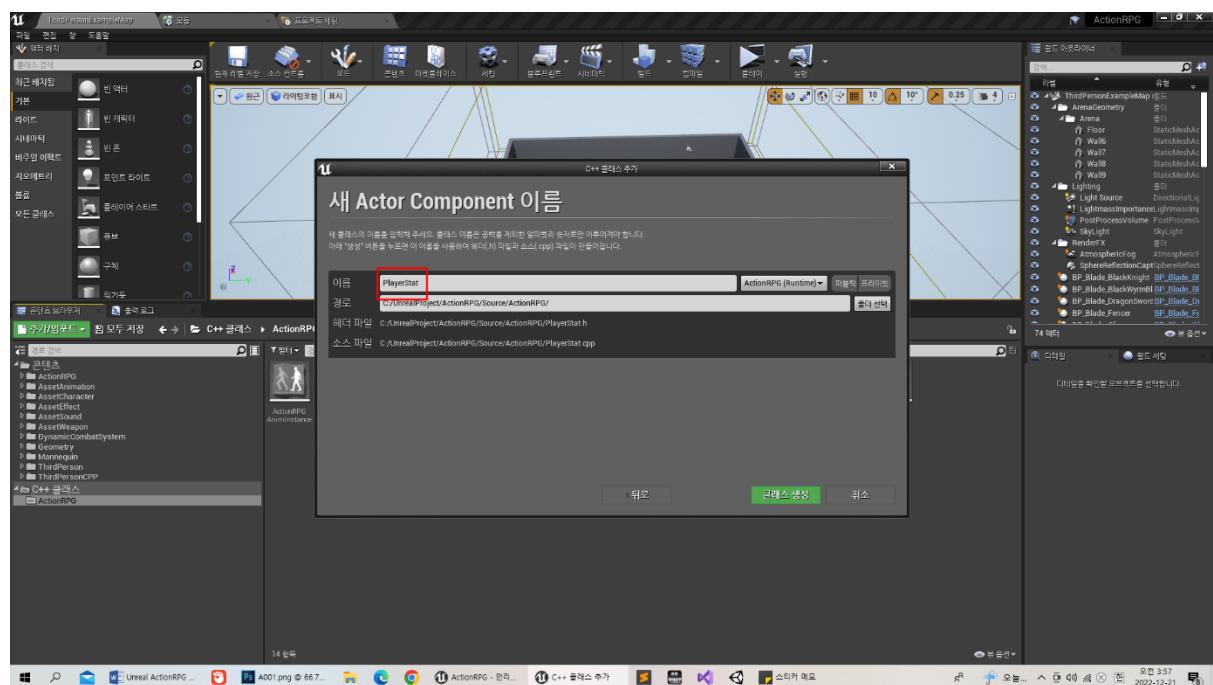
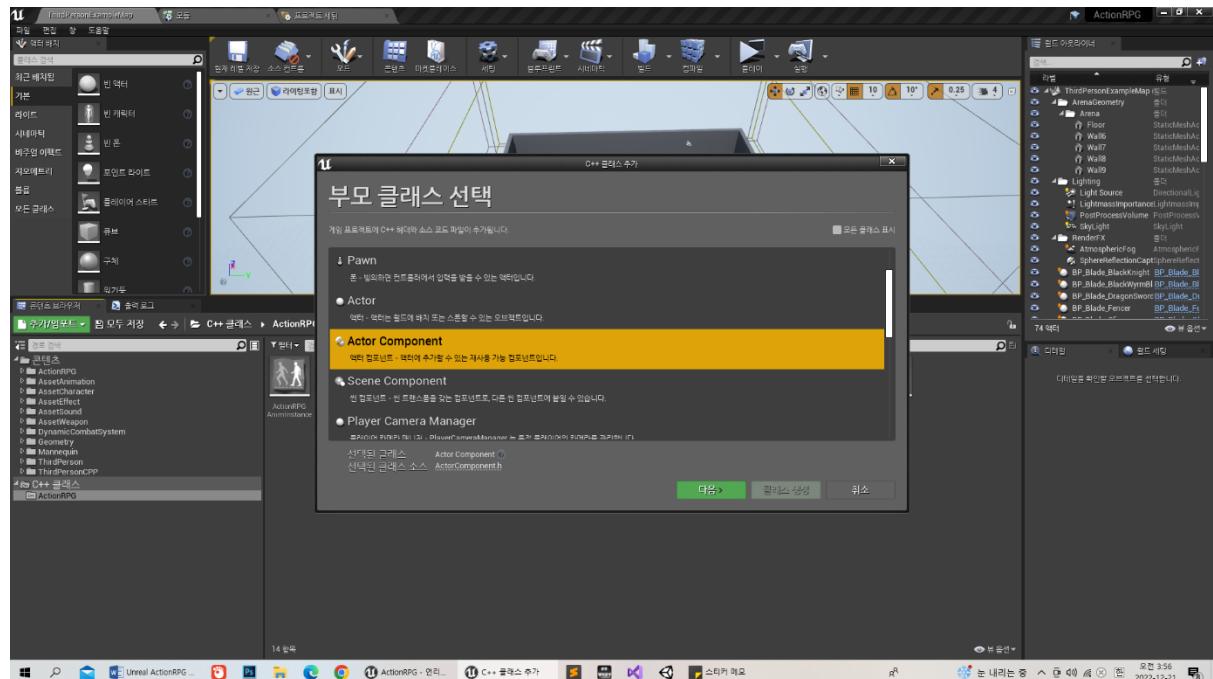
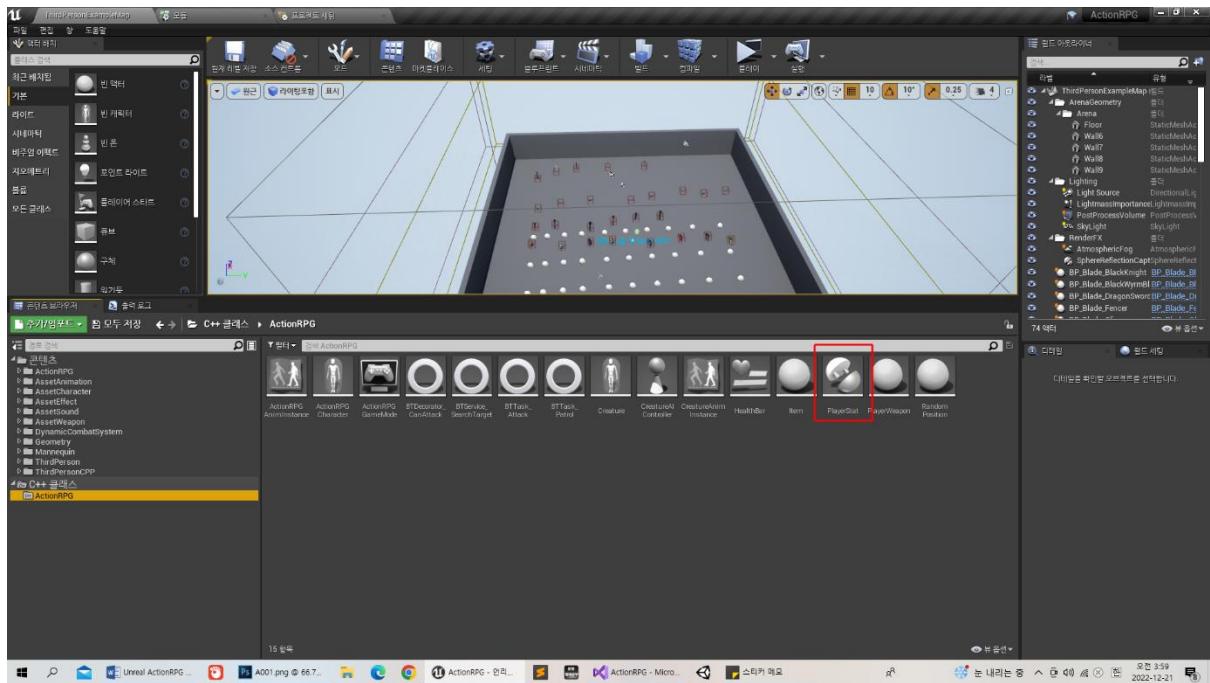


수치값을 코드에 넣는 것을 하드코딩이라고 합니다. 하드코딩을 없애주도록 합니다.

플레이어부터 해 주도록 합니다.

ActorComponent클래스를 부모 클래스로 상속받는 PlayerStat이라는 이름의 클래스를 정의해 줍니다.

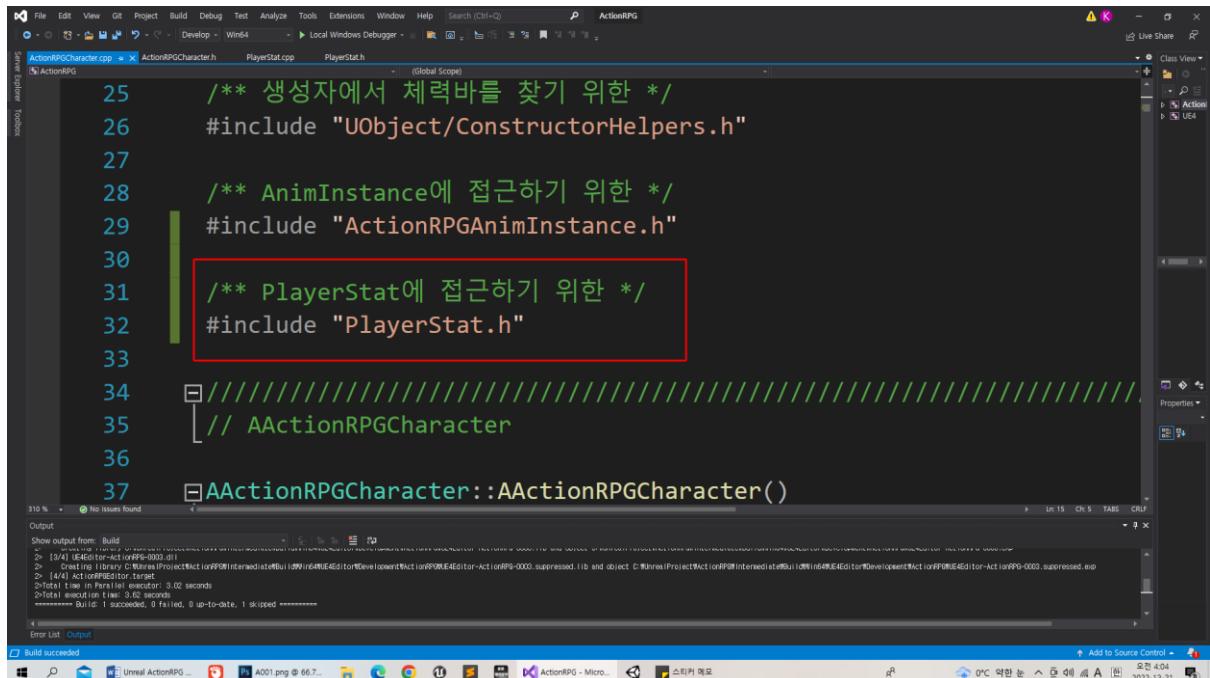




플레이어 클래스에서 생성해 주도록 합니다.

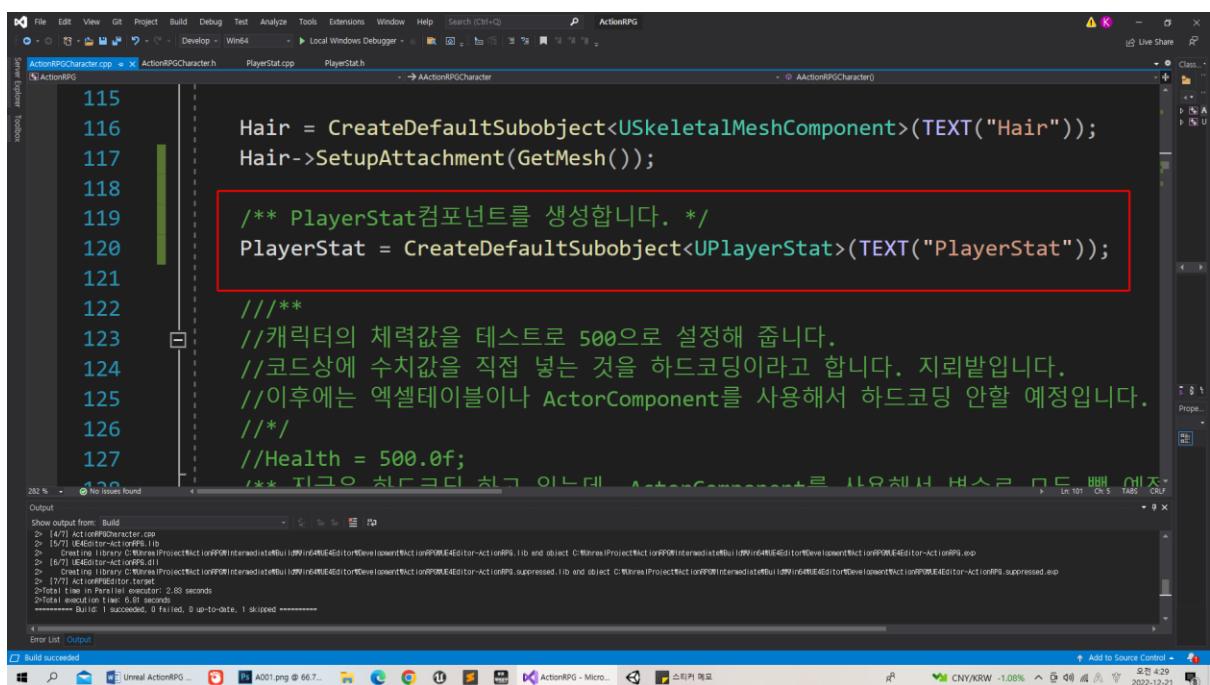
```

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()
```

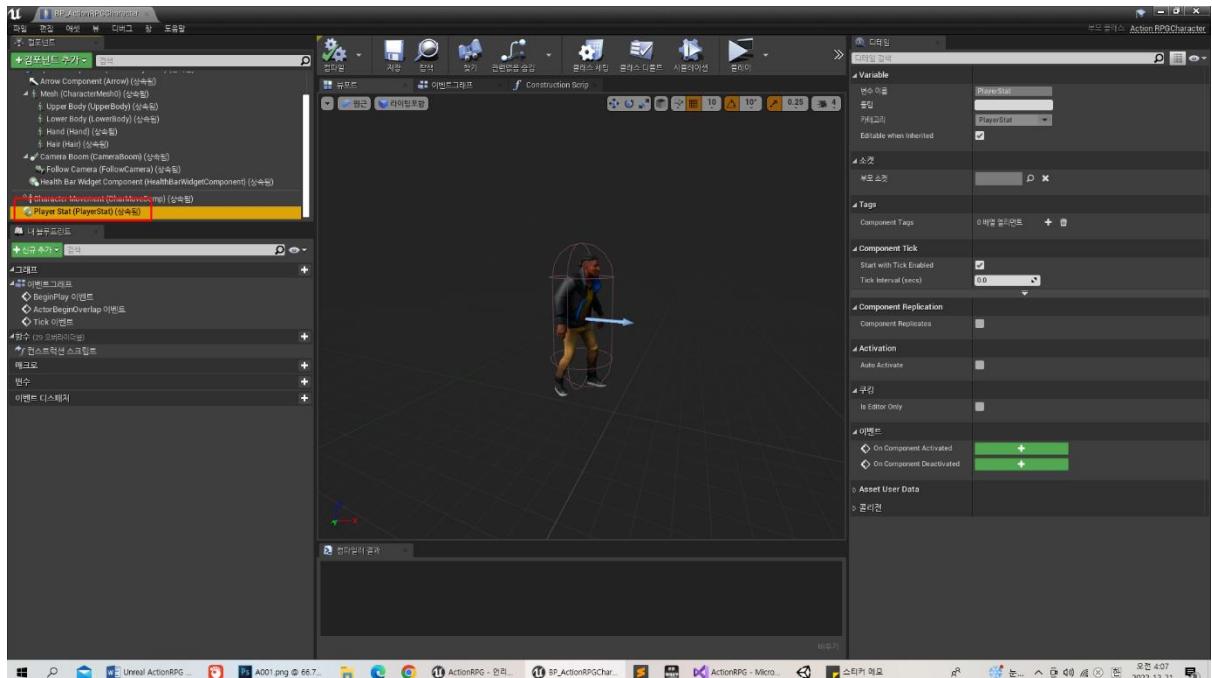
Output:
Build succeeded
2023-12-21 20:40:40



```
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

```

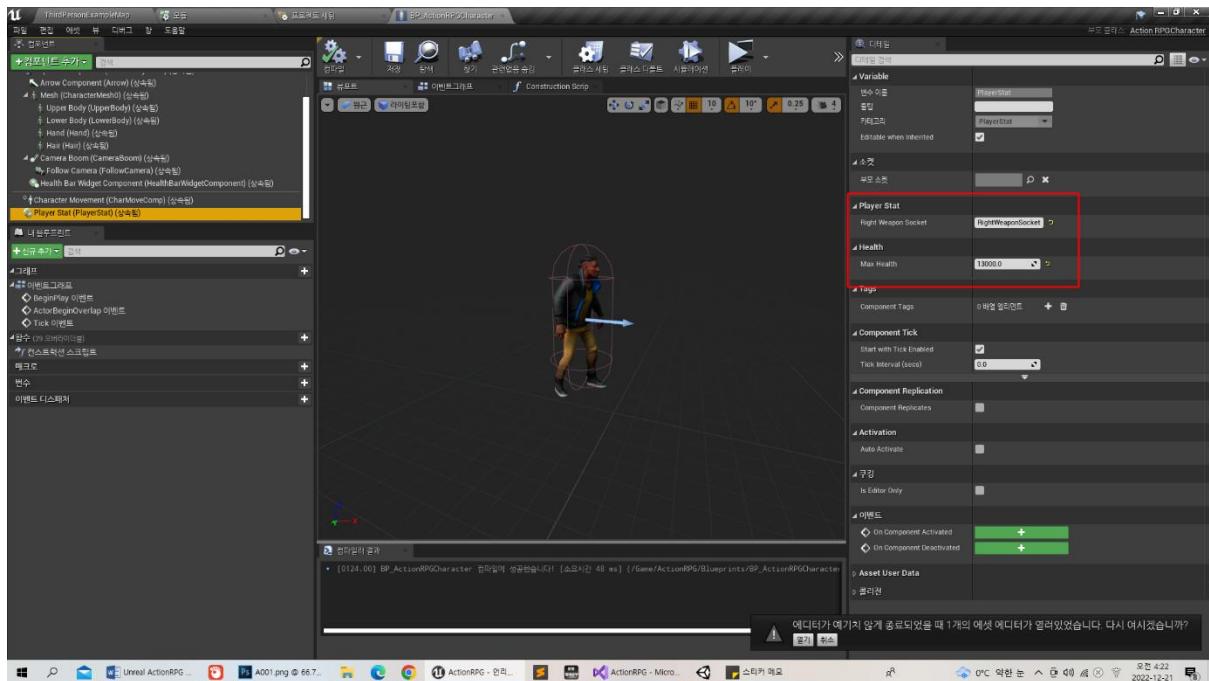
The code is a C++ class definition for PlayerStat. It includes a private section with a RightWeaponSocket variable and a MaxHealth variable. It also includes a public section with a GetMaxHealth() function.

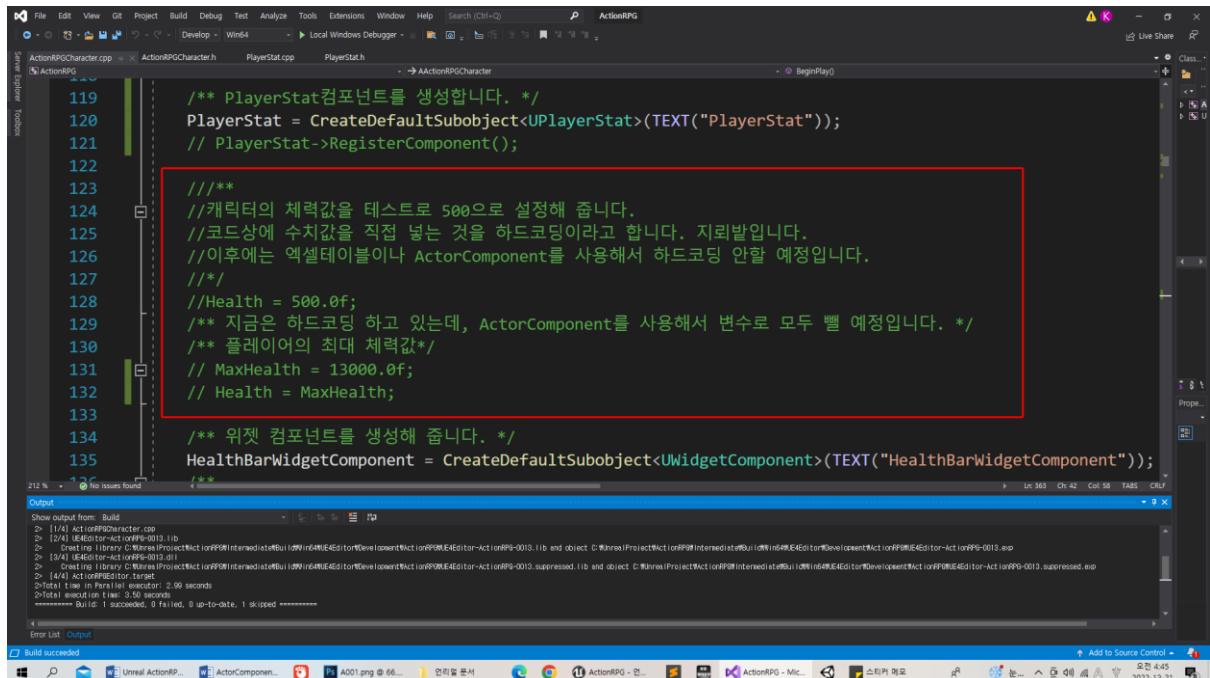
```

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

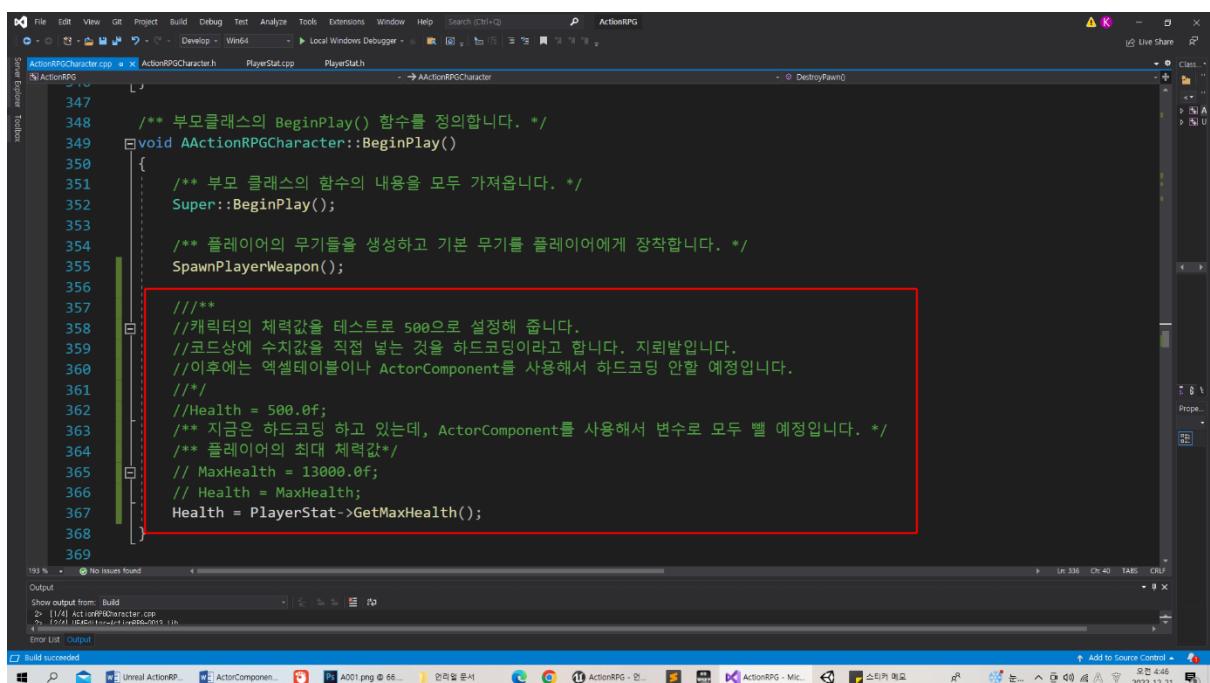
```

디테일 패널에서 확인해 봅니다.

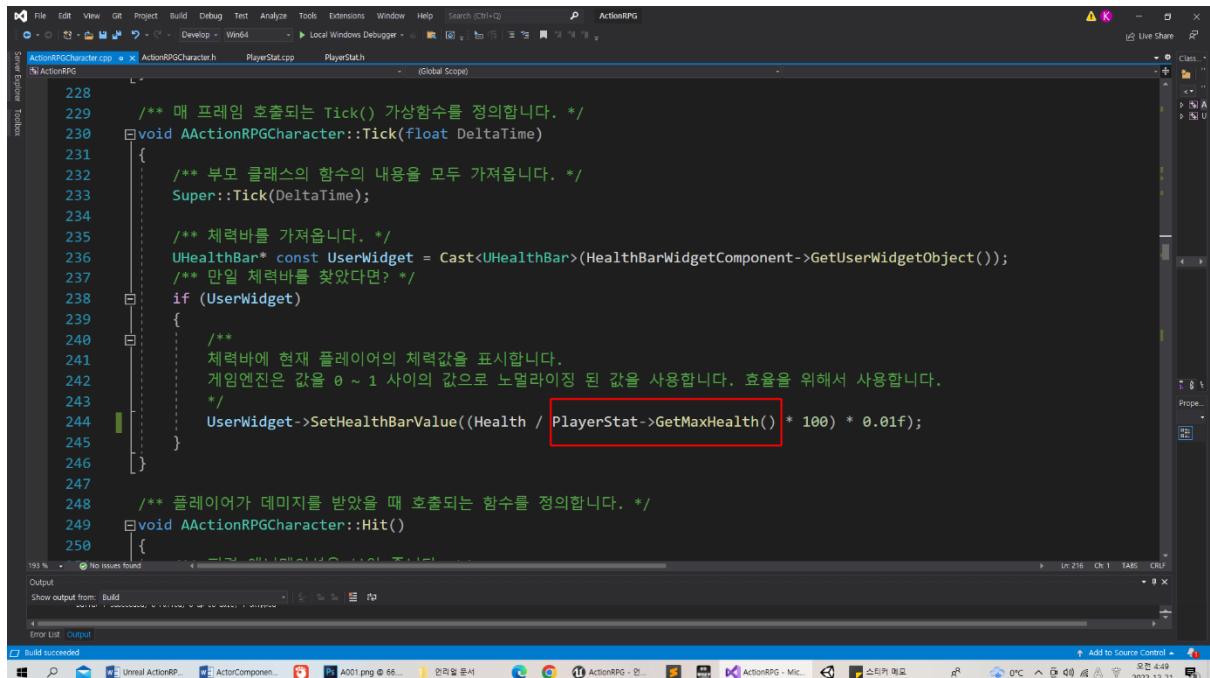




```
119     /** PlayerStat 컴포넌트를 생성합니다. */
120     PlayerStat = CreateDefaultSubobject<UPlayerStat>(TEXT("PlayerStat"));
121     // PlayerStat->RegisterComponent();
122
123     /**
124      //캐릭터의 체력값을 테스트로 500으로 설정해 줍니다.
125      //코드상에 수치값을 직접 넣는 것을 하드코딩이라고 합니다. 지뢰발입니다.
126      //이후에는 엑셀테이블이나 ActorComponent를 사용해서 하드코딩 안할 예정입니다.
127      /**
128      //Health = 500.0f;
129      /** 지금은 하드코딩 하고 있는데, ActorComponent를 사용해서 변수로 모두 뺄 예정입니다. */
130      /** 플레이어의 최대 체력값*/
131      // MaxHealth = 13000.0f;
132      // Health = MaxHealth;
133
134     /** 위젯 컴포넌트를 생성해 줍니다. */
135     HealthBarWidgetComponent = CreateDefaultSubobject<UWidgetComponent>(TEXT("HealthBarWidgetComponent"));
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
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
```

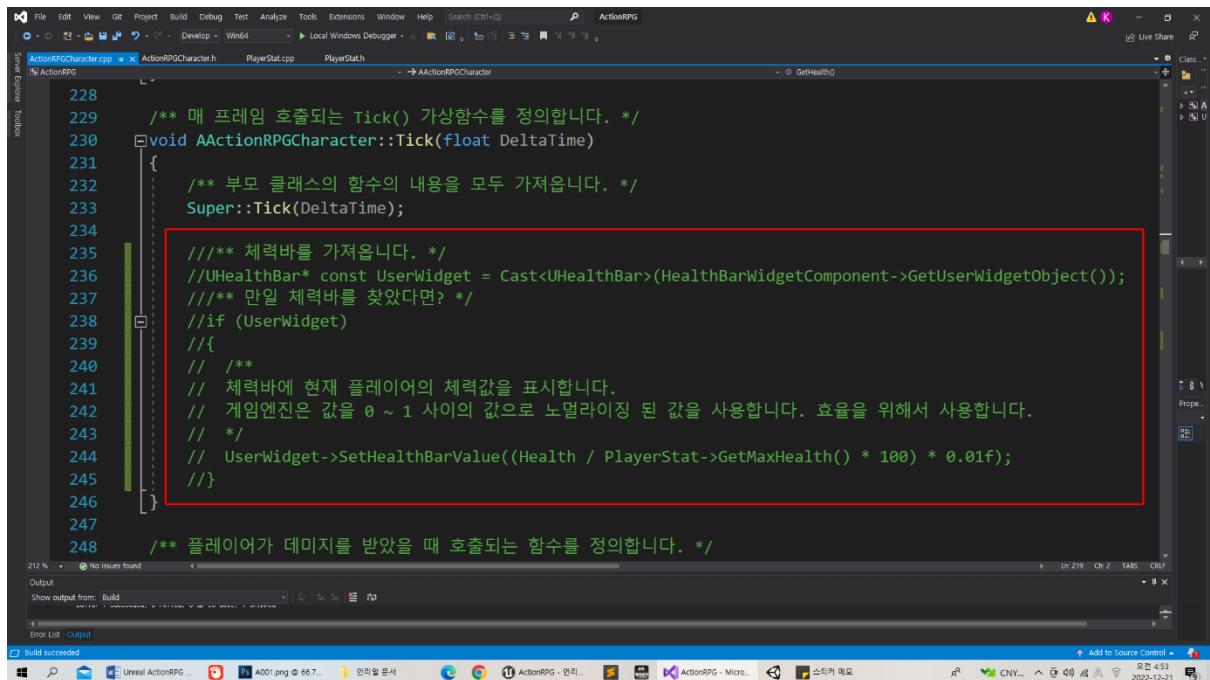


```
347     /** 부모클래스의 BeginPlay() 함수를 정의합니다. */
348     void AActionRPGCharacter::BeginPlay()
349     {
350         /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
351         Super::BeginPlay();
352
353         /** 플레이어의 무기들을 생성하고 기본 무기를 플레이어에게 장착합니다. */
354         SpawnPlayerWeapon();
355
356         /**
357  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

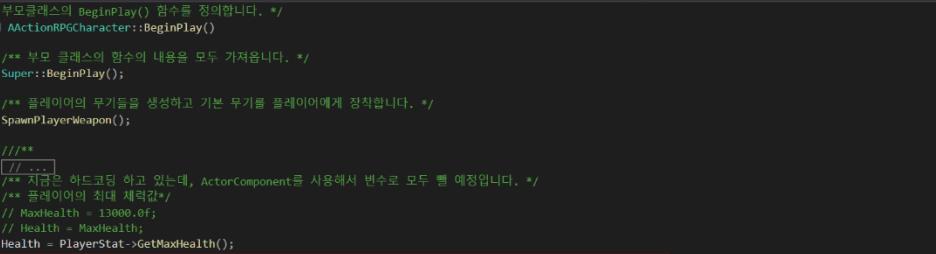


```
228  /** 매 프레임 호출되는 Tick() 가상함수를 정의합니다. */
229  void AActionRPGCharacter::Tick(float DeltaTime)
230  {
231      /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
232      Super::Tick(DeltaTime);
233
234      /** 체력바를 가져옵니다. */
235      UHealthBar* const UserWidget = Cast<UHealthBar>(HealthBarWidgetComponent->GetUserWidgetObject());
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  }
```

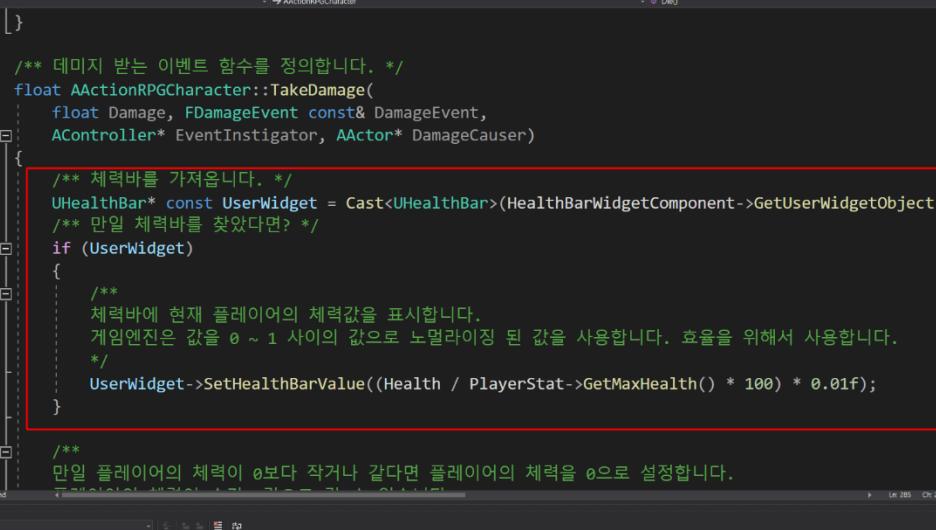
Tick() 함수 안에서 처리하면 부하가 많이 걸립니다. TakeDamage 함수안에서 처리해 주도록 합니다.



```
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
```

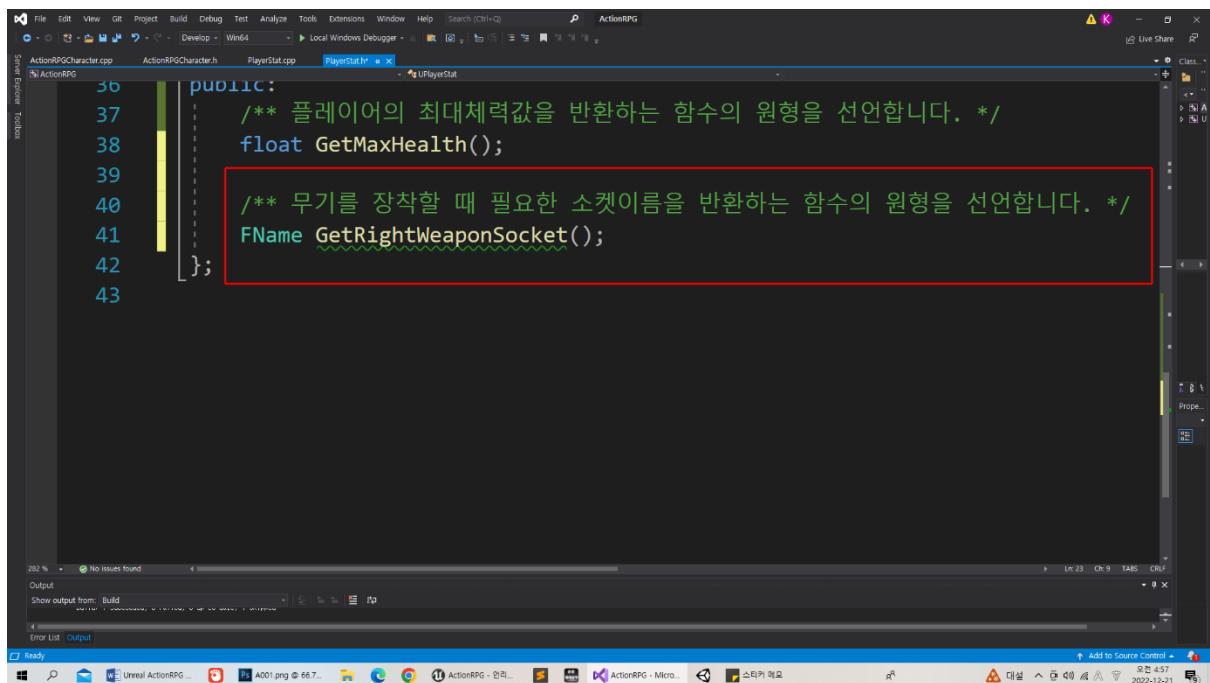


```
360     /** 부모클래스의 BeginPlay() 함수를 정의합니다. */
361     void AACTIONRPGCharacter::BeginPlay()
362     {
363         /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
364         Super::BeginPlay();
365
366         /** 플레이어의 무기들을 생성하고 기본 무기를 플레이어에게 장착합니다. */
367         SpawnPlayerWeapon();
368
369         //////
370         // ...
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     // ...
395 }
```

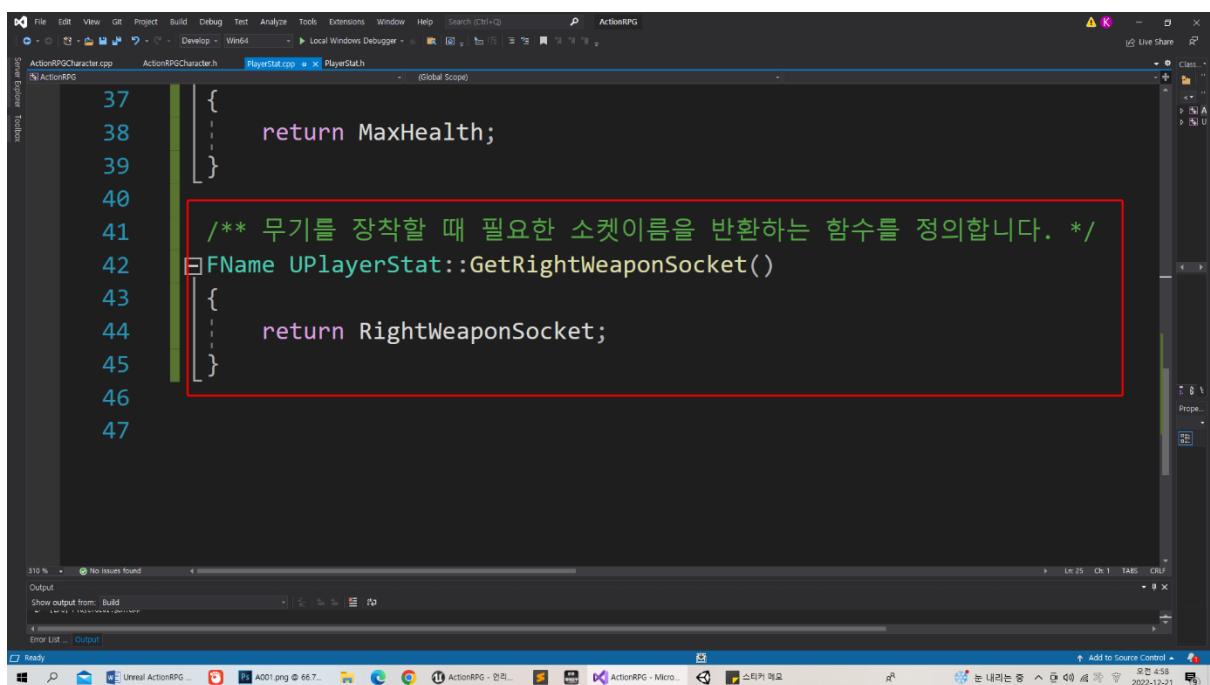


```
296     }
297
298     /** 데미지 받는 이벤트 함수를 정의합니다. */
299     float ActionRPGCharacter::TakeDamage(
300         float Damage, FDamageEvent const& DamageEvent,
301         AController* EventInstigator, AActor* DamageCauser)
302     {
303         /** 체력바를 가져옵니다. */
304         UHealthBar* const UserWidget = 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.

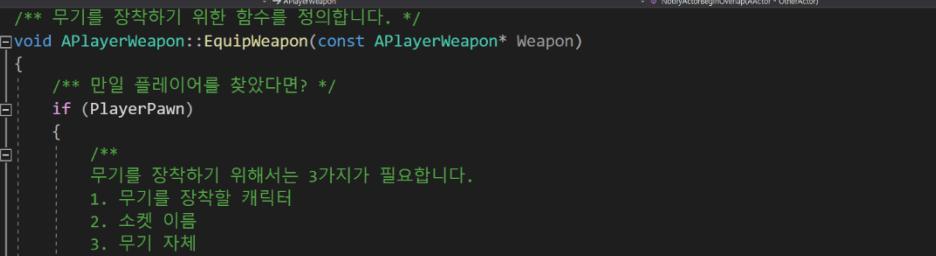


```
36     public:
37     /** 플레이어의 최대체력값을 반환하는 함수의 원형을 선언합니다. */
38     float GetMaxHealth();
39
40     /** 무기를 장착할 때 필요한 소켓이름을 반환하는 함수의 원형을 선언합니다. */
41     FName GetRightWeaponSocket();
42
43 }
```



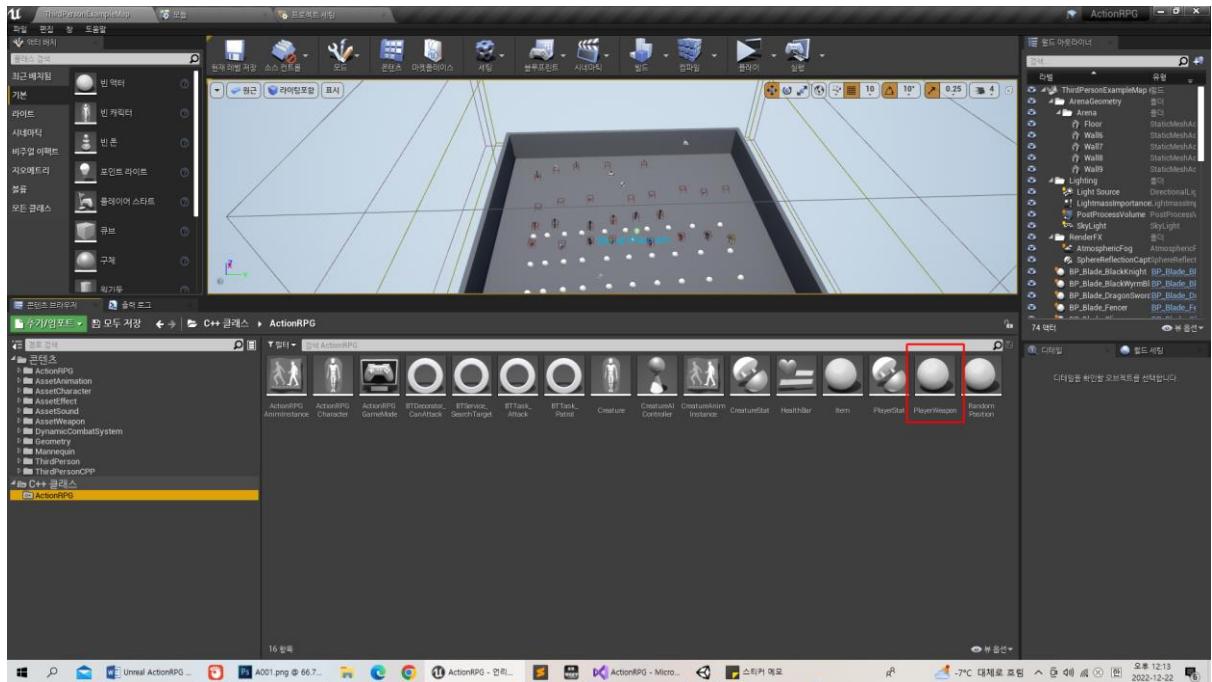
```
37     {
38     return MaxHealth;
39     }
40
41     /** 무기를 장착할 때 필요한 소켓이름을 반환하는 함수를 정의합니다. */
42     FName UPlayerStat::GetRightWeaponSocket()
43     {
44     return RightWeaponSocket;
45     }
46
47 }
```

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     }
```

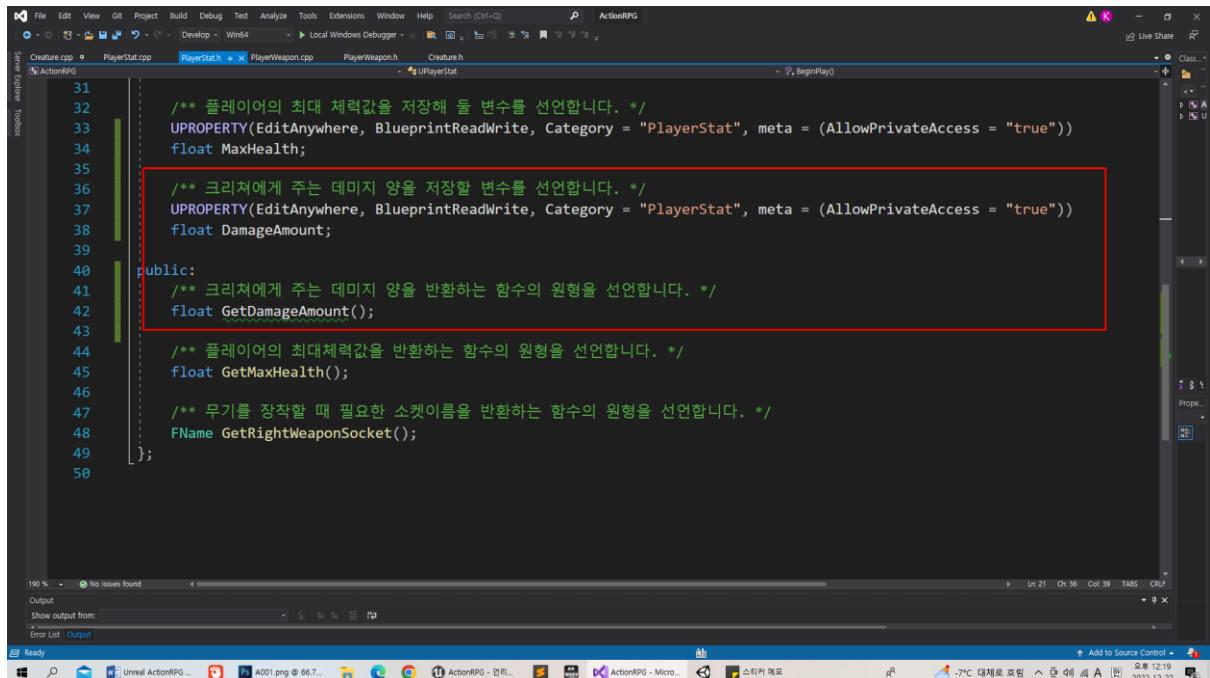
크리쳐에게 데미지를 주는 값을 적용해 주도록 합니다.



```

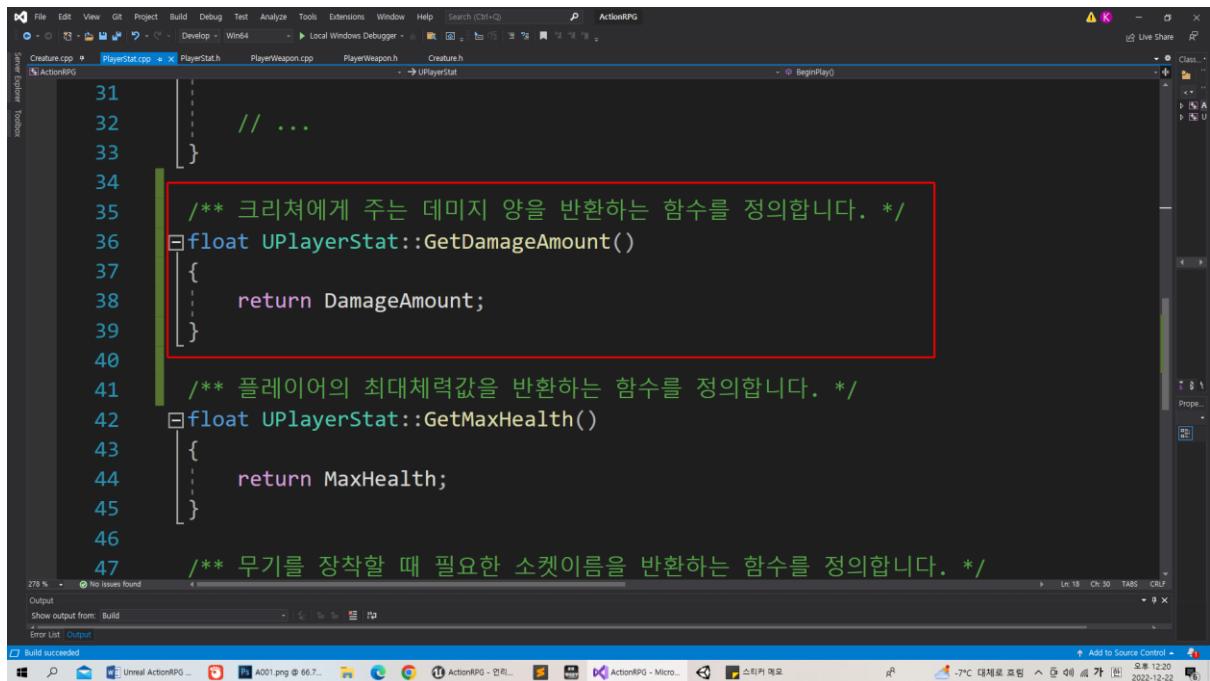
Creature.cpp  PlayerWeapon.cpp  PlayerWeapon.h  Creature.h
Creature.h
83  /** 충돌을 처리하는 가상함수를 정의합니다. */
84  void APlayerWeapon::NotifyActorBeginOverlap(AActor* OtherActor)
85  {
86      /** 부모 클래스의 함수의 내용을 모두 가져옵니다. */
87      Super::NotifyActorBeginOverlap(OtherActor);
88
89      /**
90      플레이어가 가지고 있는 무기입니다. 플레이어랑은 충돌체크를 하지 않습니다.
91      플레이어가 가지고 있는 무기이니 플레이어라면 함수를 빠져 나갑니다.
92      StaticClass() : UCLASS를 반환합니다.
93      */
94      if (OtherActor->IsA(AActionRPGCharacter::StaticClass()))
95      {
96          return;
97      }
98
99      /**
100      만일 충돌한 오브젝트가 있다면? */
101      if (OtherActor->IsA(AActor::StaticClass()))
102      {
103          /**
104          50의 데미지를 줍니다. 이후에는 ActorComponent에 수치값을 정의해 줄 예정입니다.
105          ApplyDamage이벤트 함수로 데미지를 주고, TakeDamage이벤트 함수로 데미지를 받습니다.
106          */
107          UGameplayStatics::ApplyDamage(OtherActor, 1100.0f, nullptr, this, UDamageType::StaticClass());
108      }
109 }

```



```
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.



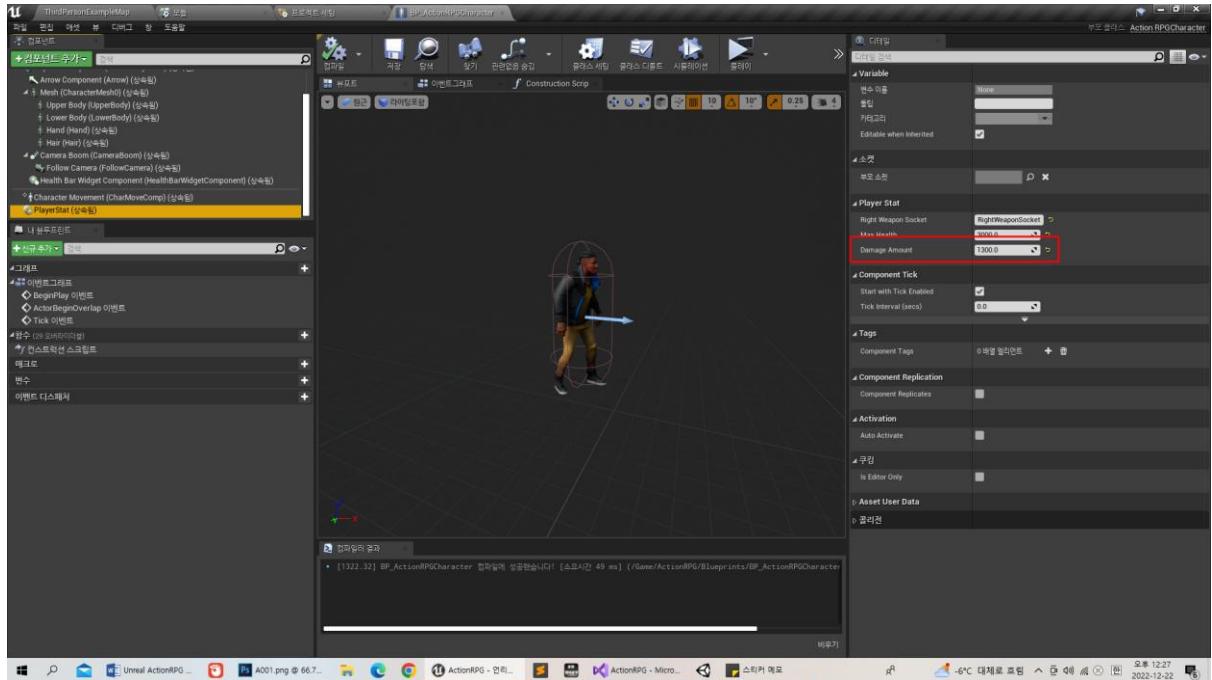
```
31
32  // ...
33 }
34
35  /** 크리쳐에게 주는 데미지 양을 반환하는 함수를 정의합니다. */
36 float UPlayerStat::GetDamageAmount()
37 {
38     return DamageAmount;
39 }
40
41  /** 플레이어의 최대체력값을 반환하는 함수를 정의합니다. */
42 float UPlayerStat::GetMaxHealth()
43 {
44     return MaxHealth;
45 }
46
47  /** 무기를 장착할 때 필요한 소켓이름을 반환하는 함수를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

PlayerWeapon 클래스에서 수정해 주도록 합니다.

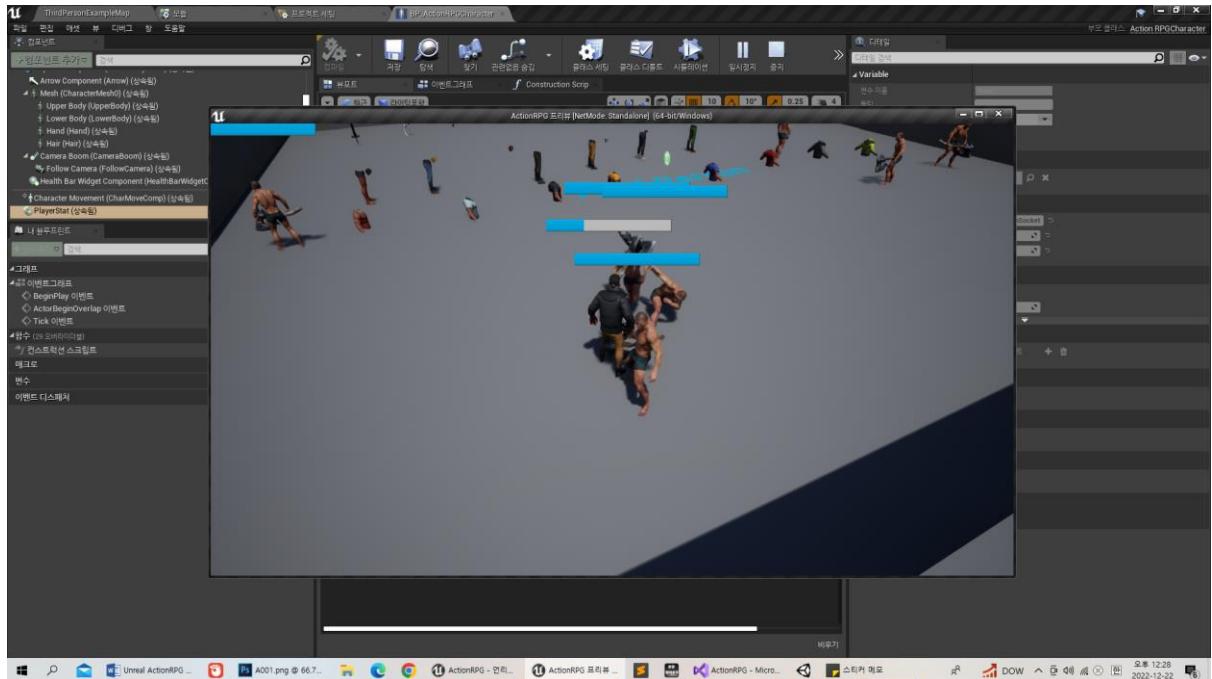
Screenshot of the Unreal Engine 4 Editor showing the PlayerWeapon.h source code. The code handles collision detection and damage application. A red box highlights the section where damage is applied to other actors:

```

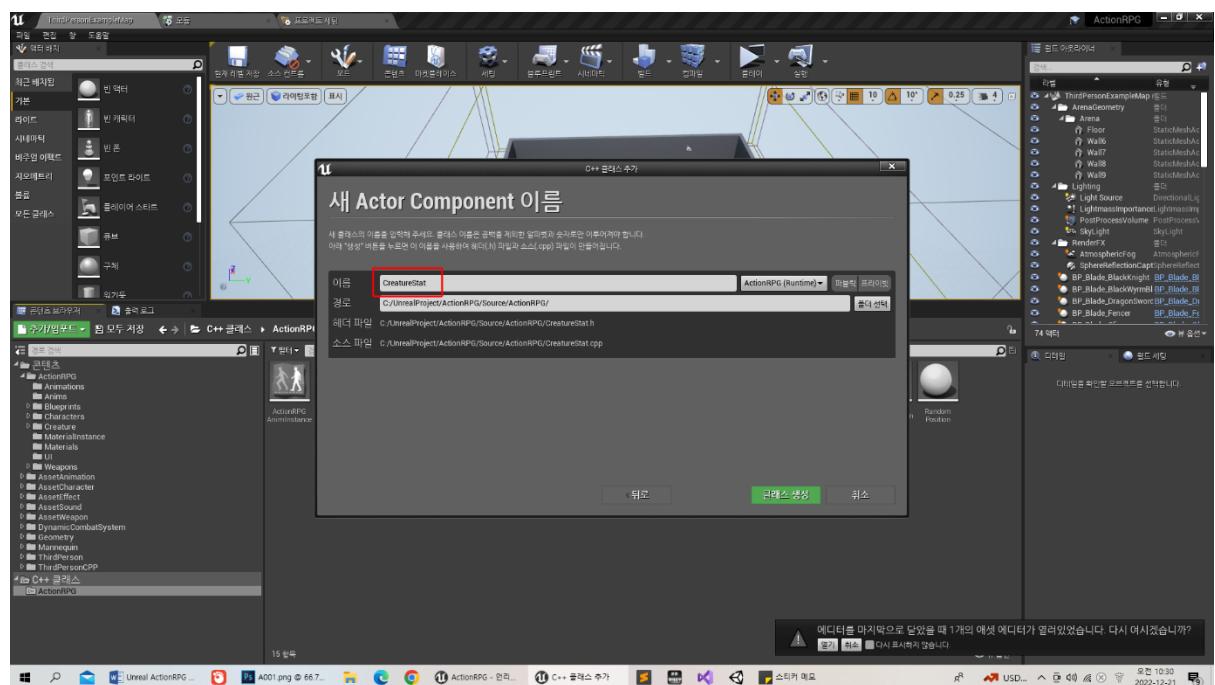
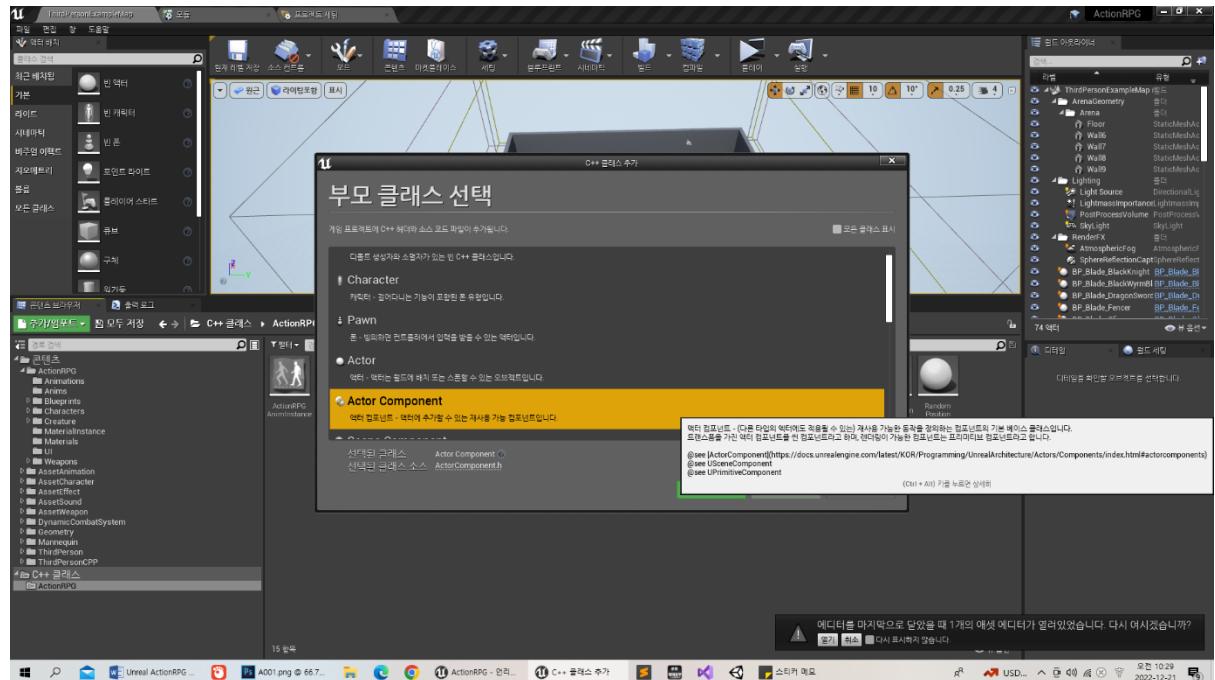
83  /** 충돌을 처리하는 기상함수를 정의합니다. */
84  void APPlayerWeapon::NotifyActorBeginOverlap(AActor* OtherActor)
85  {
86      /* 부모 클래스의 함수의 내용을 모두 가져옵니다. */
87      Super::NotifyActorBeginOverlap(OtherActor);
88
89      /**
90      플레이어가 가지고 있는 무기입니다. 플레이어랑은 충돌체크를 하지 않습니다.
91      플레이어가 가지고 있는 무기이니 플레이어라면 함수를 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         /** 플레이어를 찾습니다.*/
108         AActionRPGCharacter* PlayerAvatar = Cast<AActionRPGCharacter>(UGameplayStatics::GetPlayerCharacter(GetWorld(), 0));
109         /** 데미지 양을 시정해 줍니다.*/
110         float DamageAmount = PlayerAvatar->PlayerStat->GetDamageAmount();
111         UGameplayStatics::ApplyDamage(OtherActor, DamageAmount, nullptr, this, UDamageType::StaticClass());
112     }
113
114 }
115
116 /**
117 * bIsAttack 블리언 변수를 반환하는 함수를 정의합니다.
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
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
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
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
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
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
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
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
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
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
878
879
879
880
881
882
883
884
885
886
887
887
888
889
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
914
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1580
1581
1581
1582
1582
1583
1583
1584
1584
1585
1585
1586
1586
1587
1587
1588
1588
1589
1589
1590
1590
1591
1591
1592
1592
1593
1593
1594
1594
1595
1595
1596
1596
1597
1597
1598
1598
1599
1599
1600
1600
1601
1601
1602
1602
1603
1603
1604
1604
1605
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1610
1611
1611
1612
1612
1613
1613
1614
1614
1615
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1620
1621
1621
1622
1622
1623
1623
1624
1624
1625
1625
1626
1626
1627
1627
1628
1628
1629
1629
1630
1630
1631
1631
1632
1632
1633
1633
1634
1634
1635
1635
1636
1636
1637
1637
1638
1638
1639
1639
1640
1640
1641
1641
1642
1642
1643
1643
1644
1644
1645
1645
1646
1646
1647
1647
1648
1648
1649
1649
1650
1650
1651
1651
1652
1652
1653
```

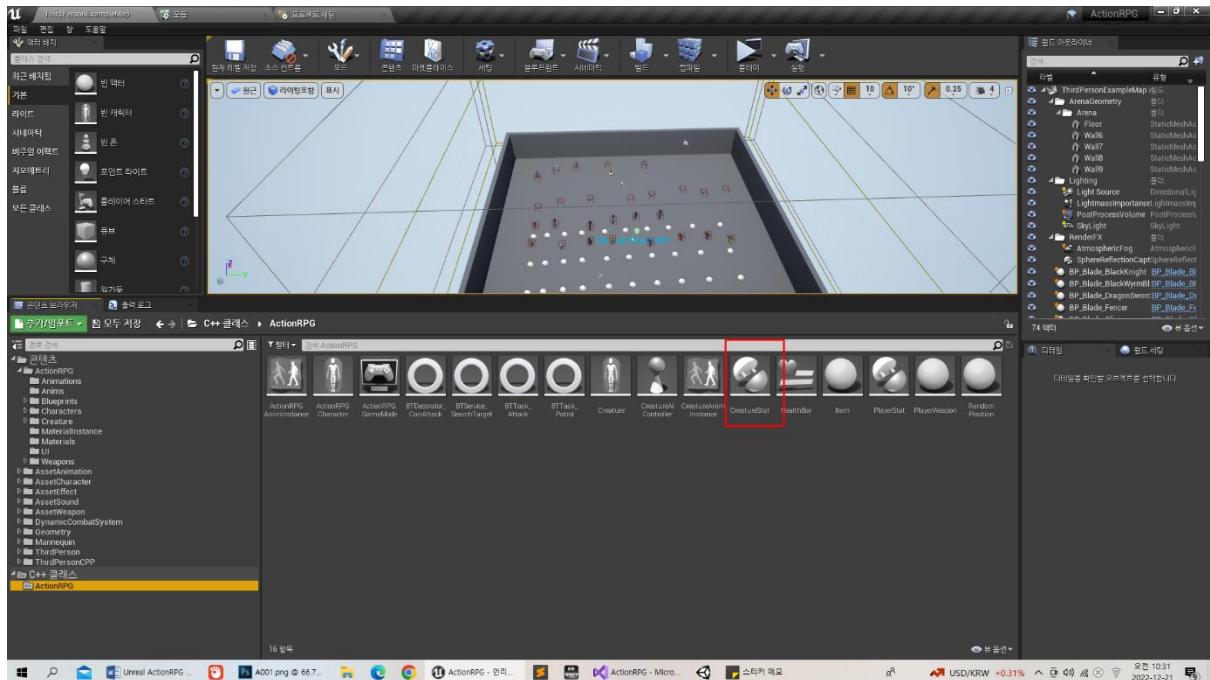


플레이를 해서 결과를 확인합니다.



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





생성자 함수에서 생성해 주도록 합니다.

```

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+F) ActionRPG

Creature.cpp Creature.h CreatureStat.cpp CreatureStat.h ActionRPGCharacter.cpp ActionRPGCharacter.h PlayerStat.cpp PlayerStat.h

76  /** BehaviorTree를 저장해 줄 변수를 선언합니다. */
77  UPROPERTY(EditAnywhere, Category = "Behavior", meta = (AllowPrivateAccess = true))
78  // UPROPERTY(EditDefaultOnly, Category = "Behavior", meta = (AllowPrivateAccess = true))
79  class UBehaviorTree* CreatureBehavior;
80
81  public:
82  /** CreatureStat을 저장할 변수를 선언합니다. */
83  UPROPERTY(EditDefaultsOnly, Category = "PlayerStat", meta = (AllowPrivateAccess = true))
84  class UCreatureStat* CreatureStat;
85
86  /** BehaviorTree를 반환하는 함수의 원형을 선언합니다. */
87  class UBehaviorTree* GetCreatureBehavior();
88
89
90  /** 게임모드를 반환하는 함수의 원형을 선언합니다. */
91  class AActionRPGGameMode* GetGameMode();
92
93  /** 충돌체크를 하는 함수의 원형을 선언합니다. */
94  virtual void NotifyActorBeginOverlap(AActor* OtherActor) override;

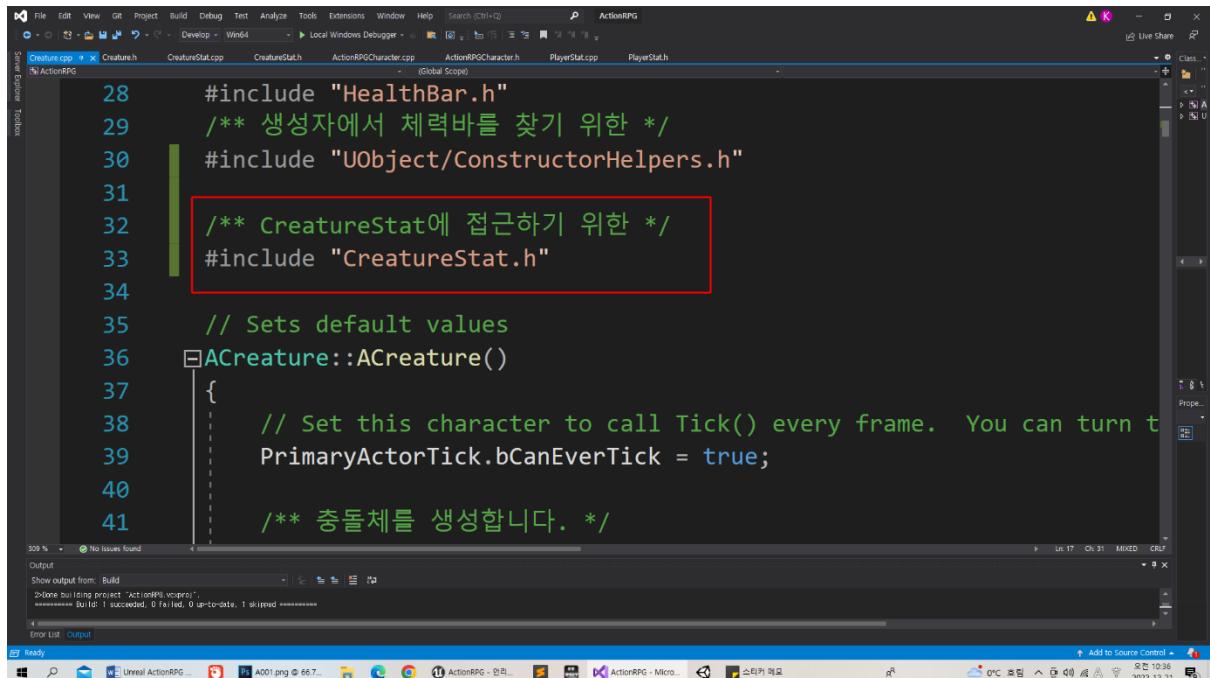
```

Output

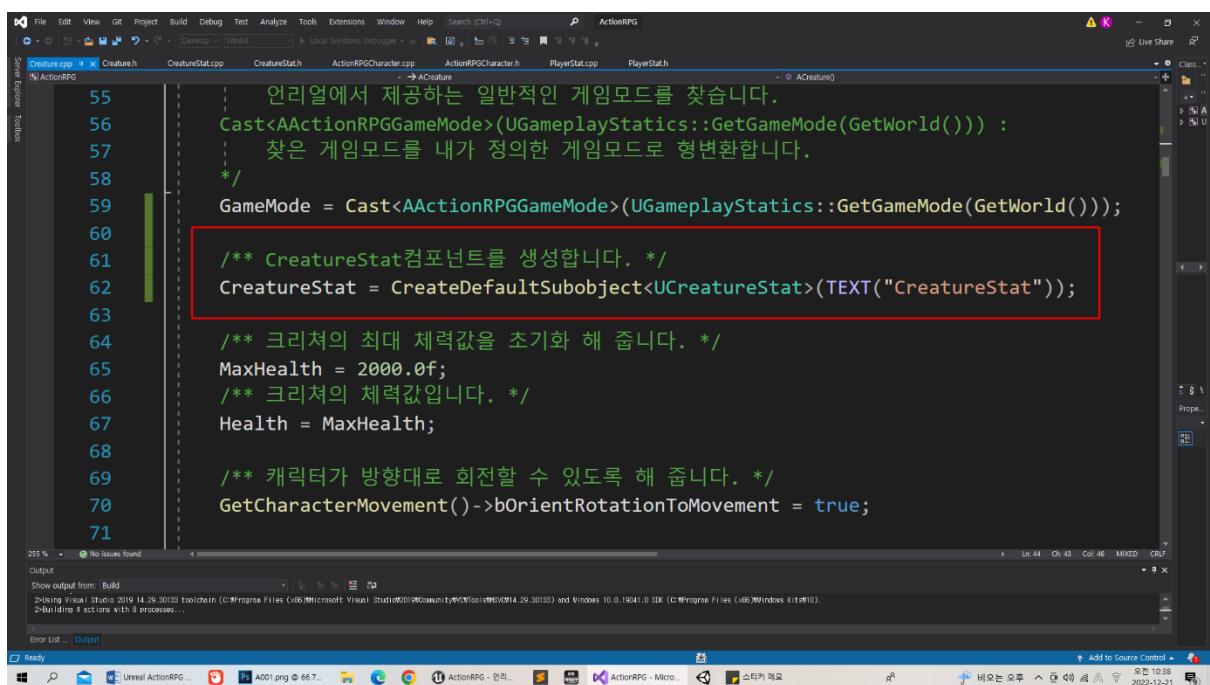
```

2022-12-21 10:36 2022-12-21 10:36
Build succeeded

```

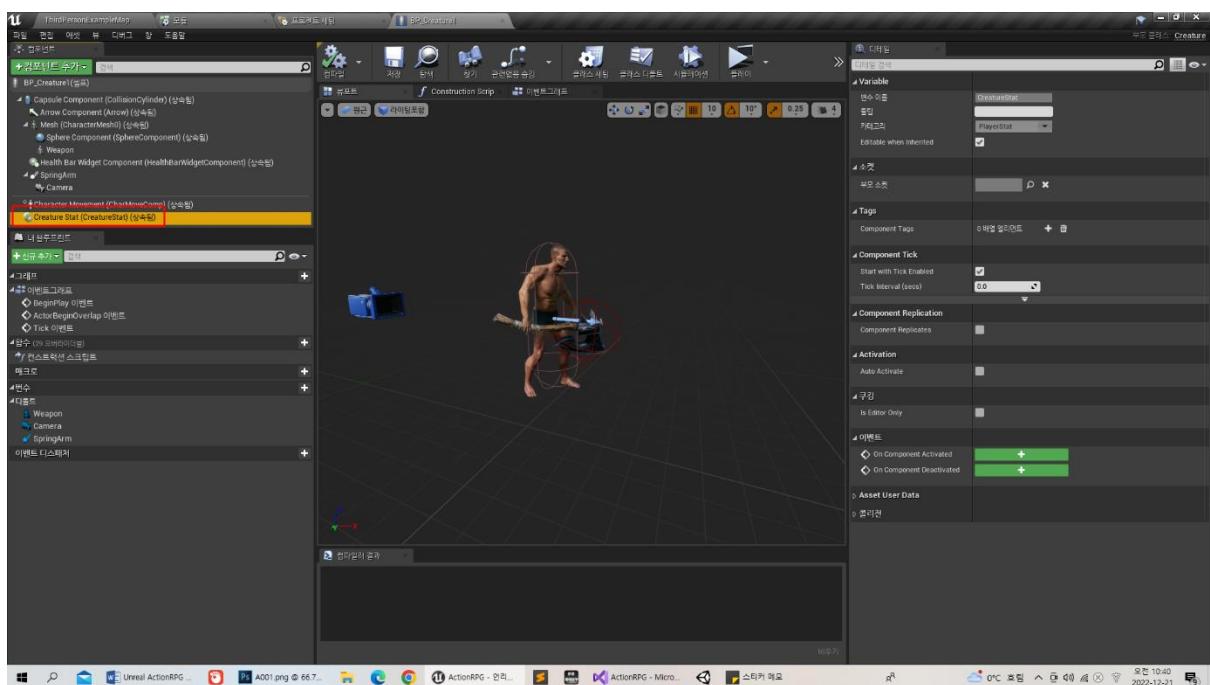
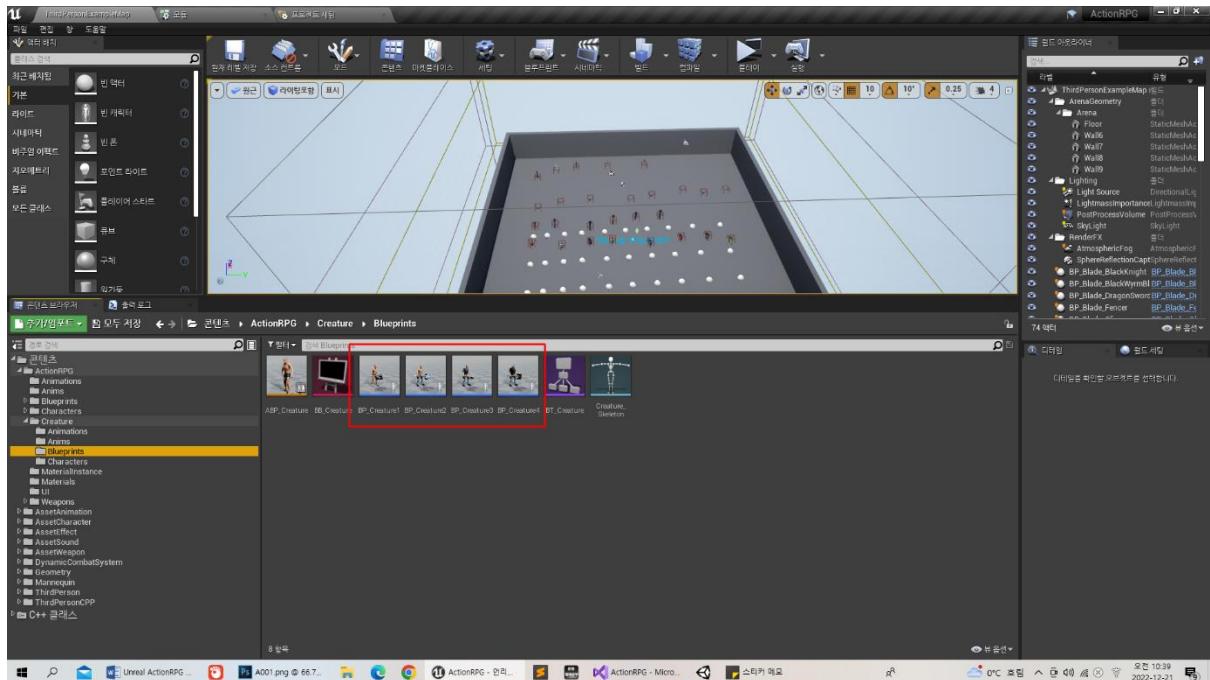


```
28     #include "HealthBar.h"
29     /** 생성자에서 체력바를 찾기 위한 */
30     #include "UObject/ConstructorHelpers.h"
31
32     /** CreatureStat에 접근하기 위한 */
33     #include "CreatureStat.h"
34
35     // Sets default values
36     ACreature::ACreature()
37     {
38         // Set this character to call Tick()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

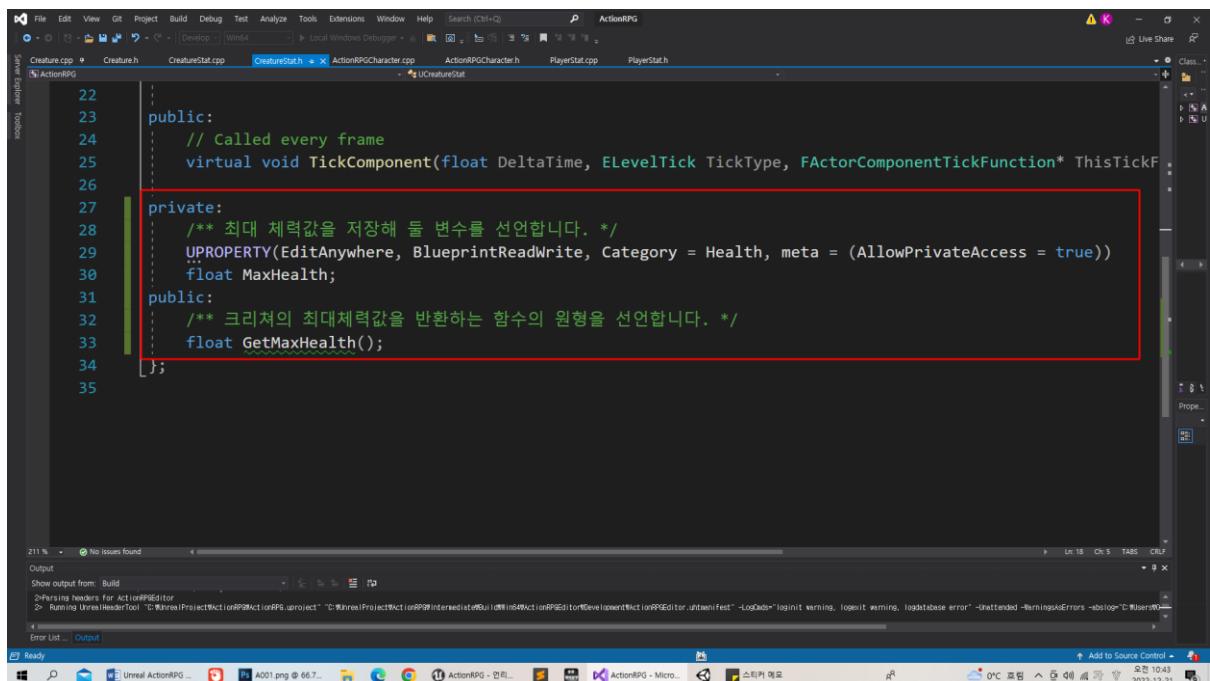


```
55     언리얼에서 제공하는 일반적인 게임모드를 찾습니다.
56     Cast<AACTIONRGGameMode>(UGameplayStatics::GetGameMode(GetWorld())) :
57         찾은 게임모드를 내가 정의한 게임모드로 형변환합니다.
58     */
59     GameMode = Cast<AACTIONRGGameMode>(UGameplayStatics::GetGameMode(GetWorld()));
60
61     /** CreatureStat컴포넌트를 생성합니다. */
62     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
```

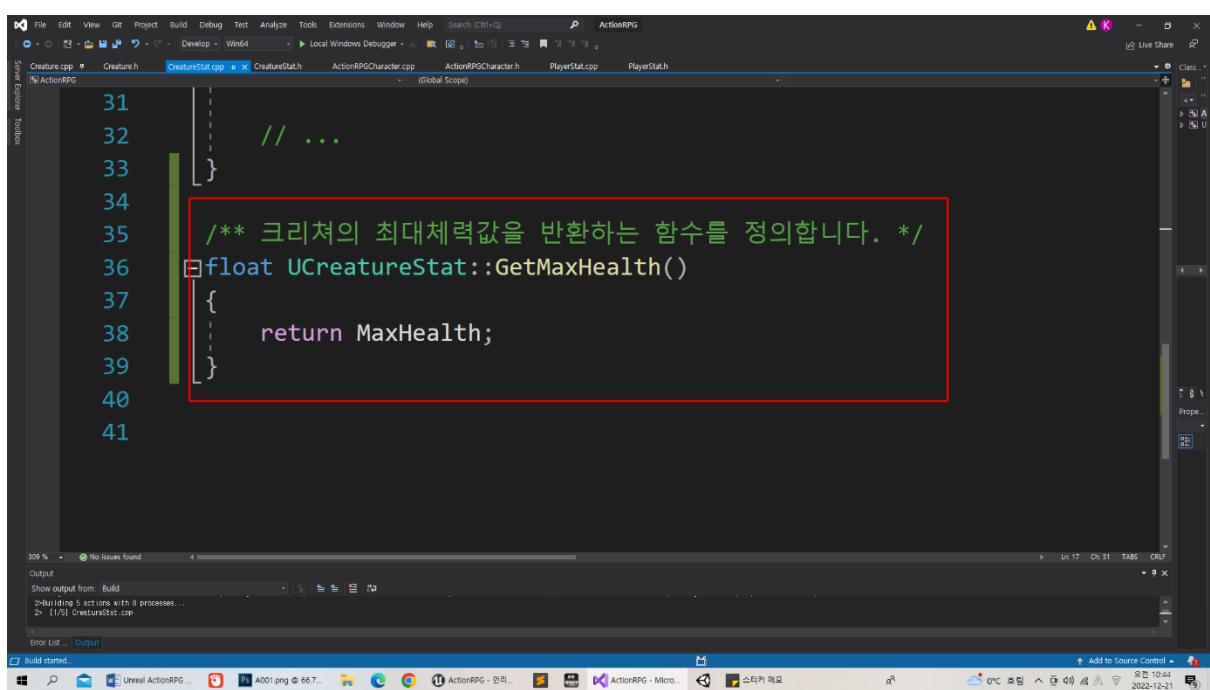
결과를 확인합니다.



크리쳐의 체력부터 적용해 주도록 합니다.

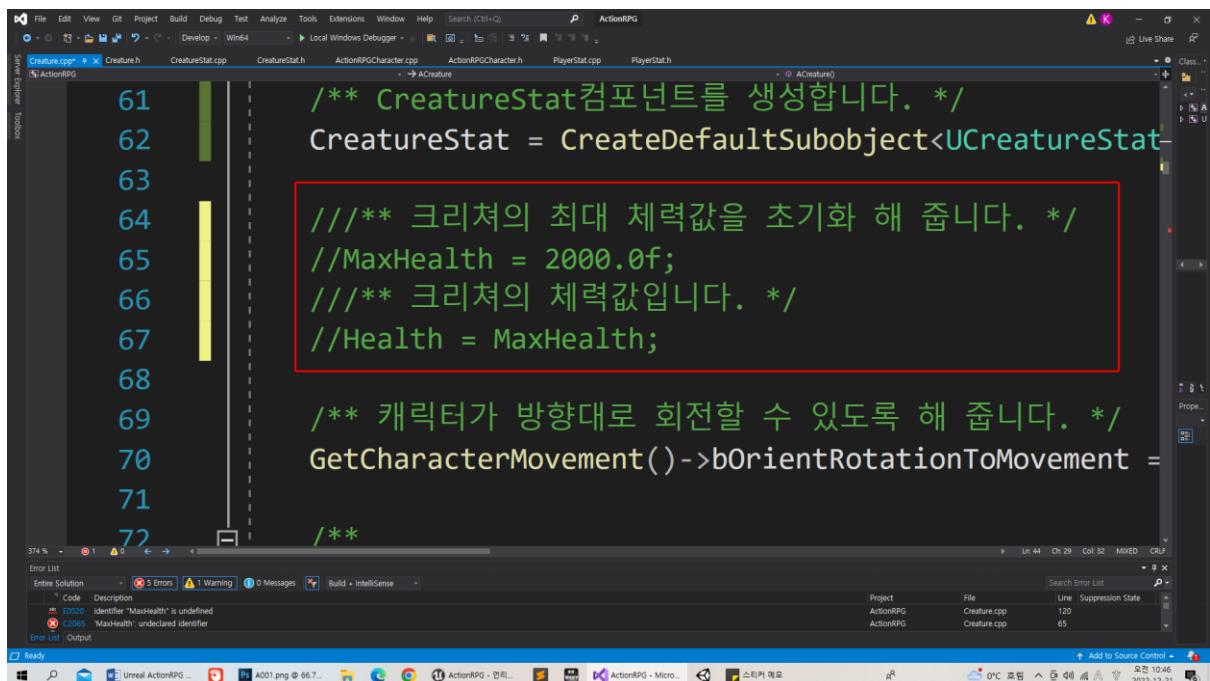


```
22
23     public:
24         // Called every frame
25         virtual void TickComponent(float DeltaTime, ELevelTick TickType, FActorComponentTickFunction* ThisTickFunction) override;
26
27     private:
28         /** 최대 체력값을 저장해 둘 변수를 선언합니다. */
29         UPROPERTY(EditAnywhere, BlueprintReadWrite,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



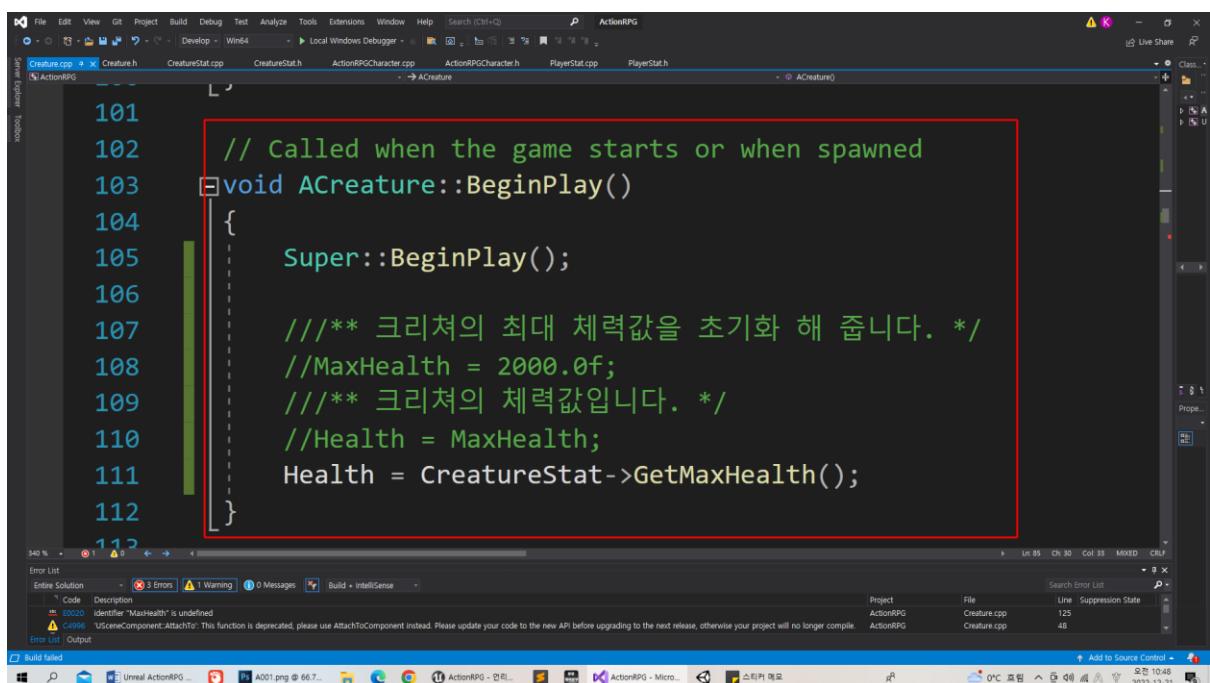
```
31
32     // ...
33
34
35     /** 크리쳐의 최대체력값을 반환하는 함수를 정의합니다. */
36     float UCreatureStat::GetMaxHealth()
37     {
38         return MaxHealth;
39     }
40
41

```

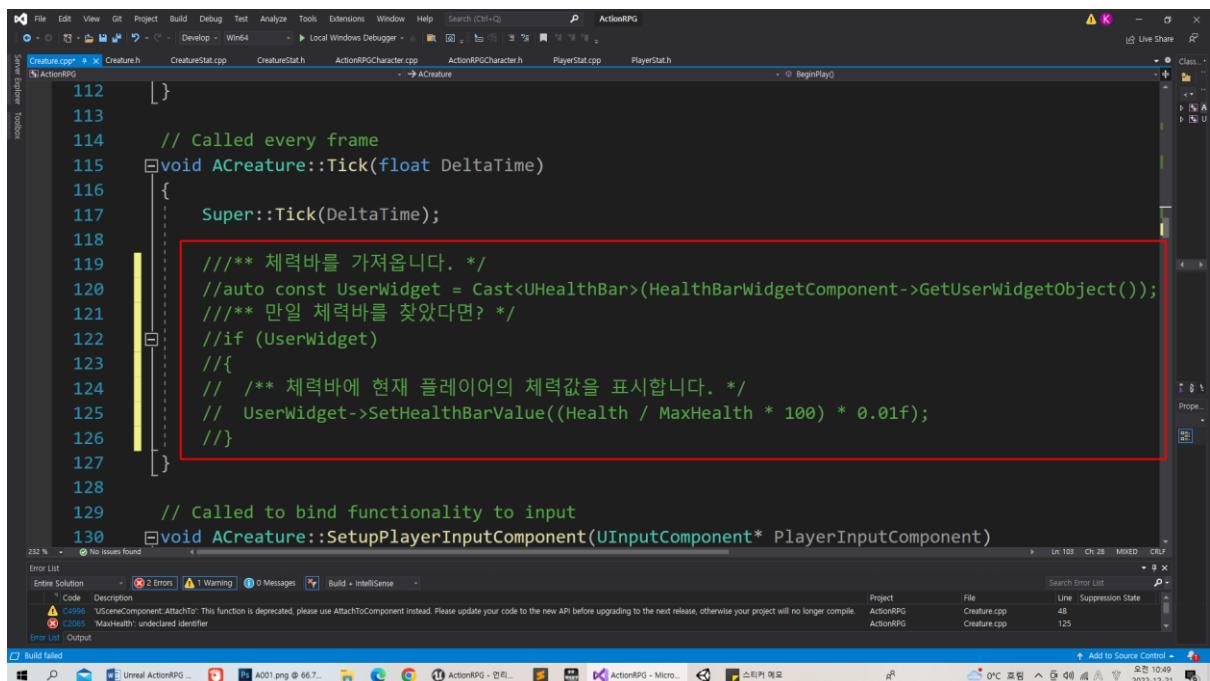


```
61  /** CreatureStat컴포넌트를 생성합니다. */
62  CreatureStat = CreateDefaultSubobject<UCreatureStat>(GetTransientByName("CreatureStat"));
63
64  // /** 크리쳐의 최대 체력값을 초기화 해 줍니다. */
65  // MaxHealth = 2000.0f;
66  // /** 크리쳐의 체력값입니다. */
67  // Health = MaxHealth;
68
69  /** 캐릭터가 방향대로 회전할 수 있도록 해 줍니다. */
70  GetCharacterMovement()->bOrientRotationToMovement = true;
71
72  /**
73
74
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
89
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
```

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



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



```
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.

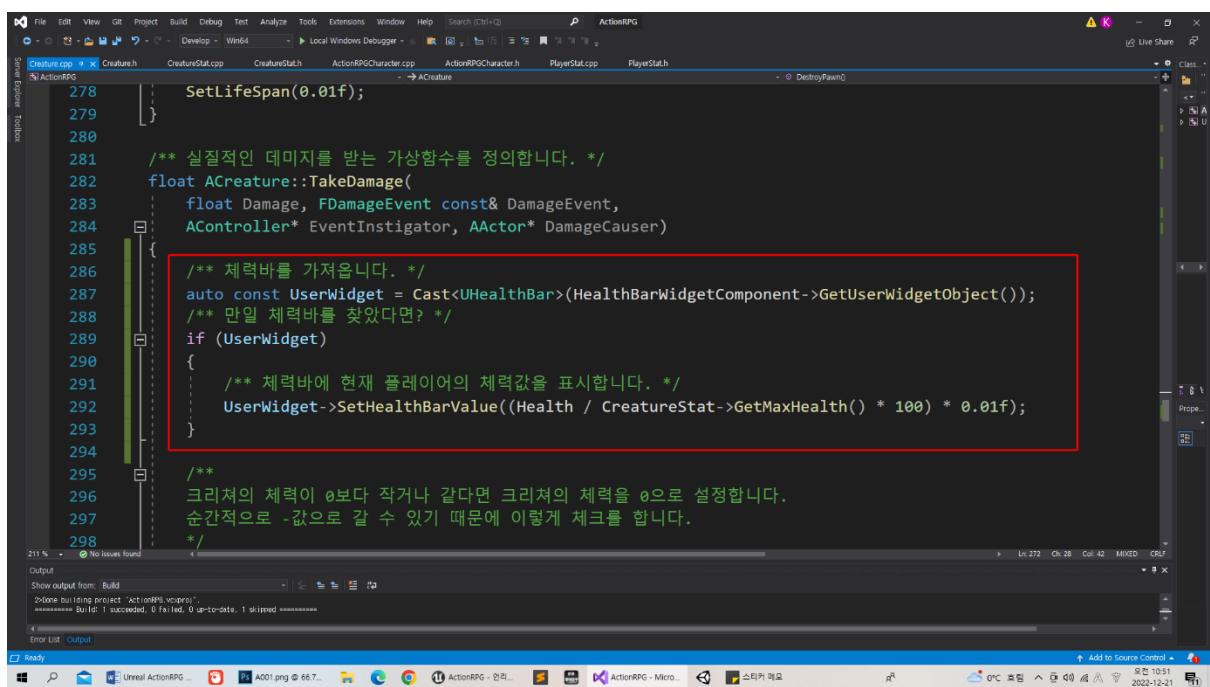
UE4-2095 'MaxHealth': undeclared identifier

Output

Build failed

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

0xC 흐름 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

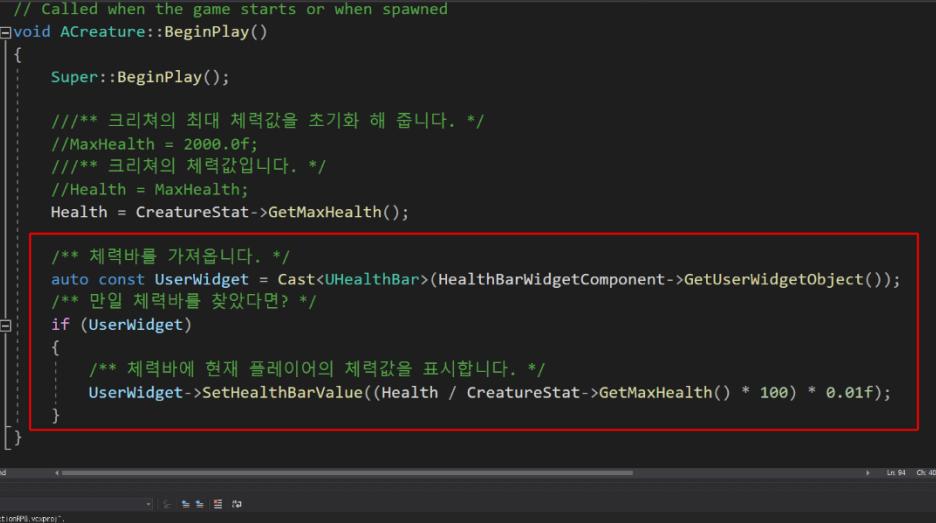
2022-12-21 10:51

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

Ready

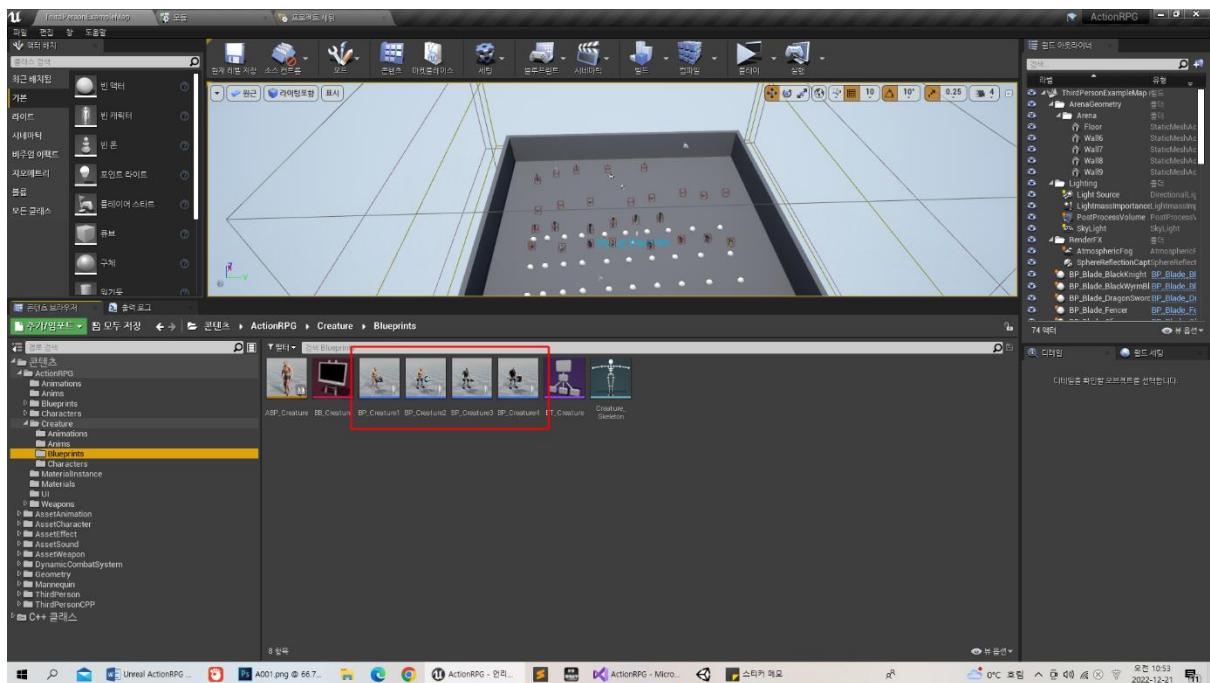
Unreal ActionRPG... A001.png @ 66.7... ActionRPG - 언리... ActionRPG - Micro... 스티커 메모

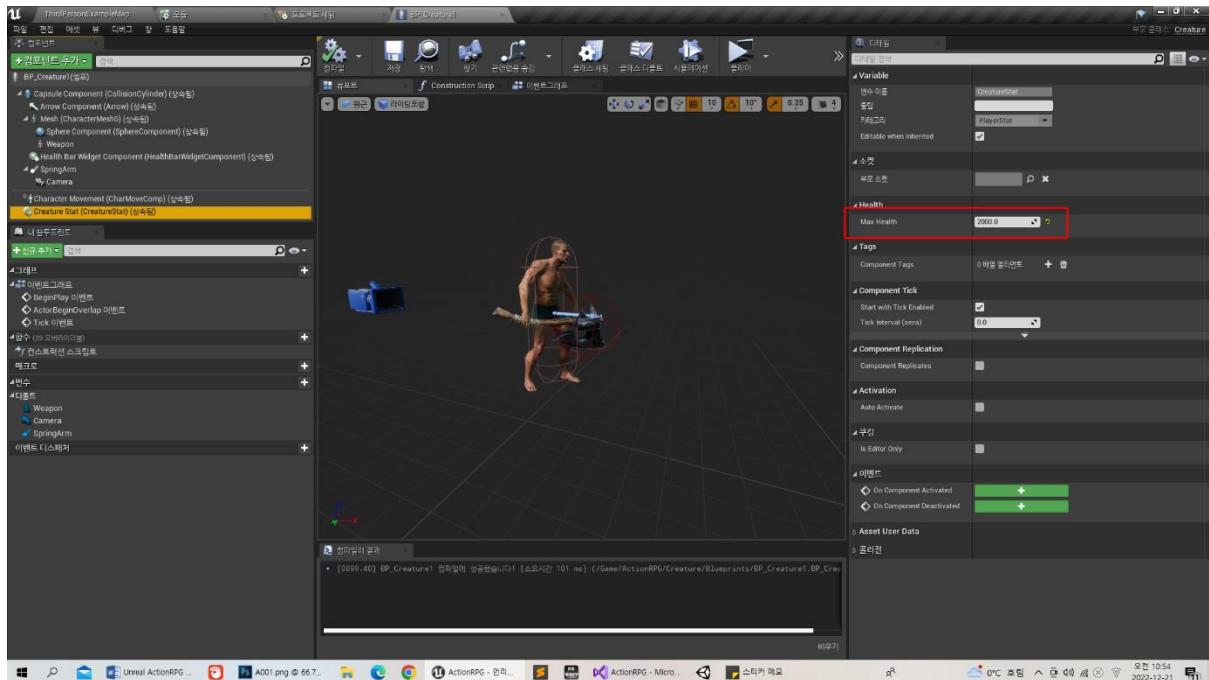
0xC 흐름 10:51 2022-12-21



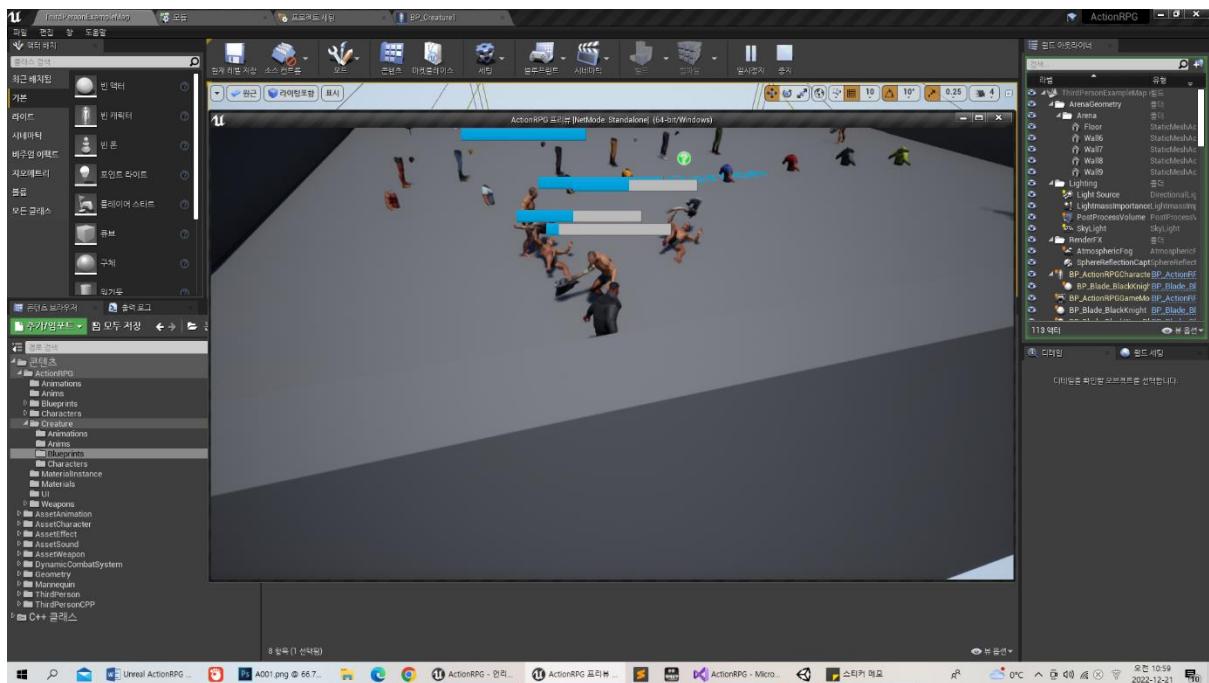
```
102 // Called when the game starts or when spawned
103 void ACreature::BeginPlay()
104 {
105     Super::BeginPlay();
106
107     /** 크리처의 최대 체력값을 초기화 해 줍니다. */
108     //MaxHealth = 2000.0f;
109     /** 크리처의 체력값입니다. */
110     //Health = MaxHealth;
111     Health = CreatureStat->GetMaxHealth();
112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

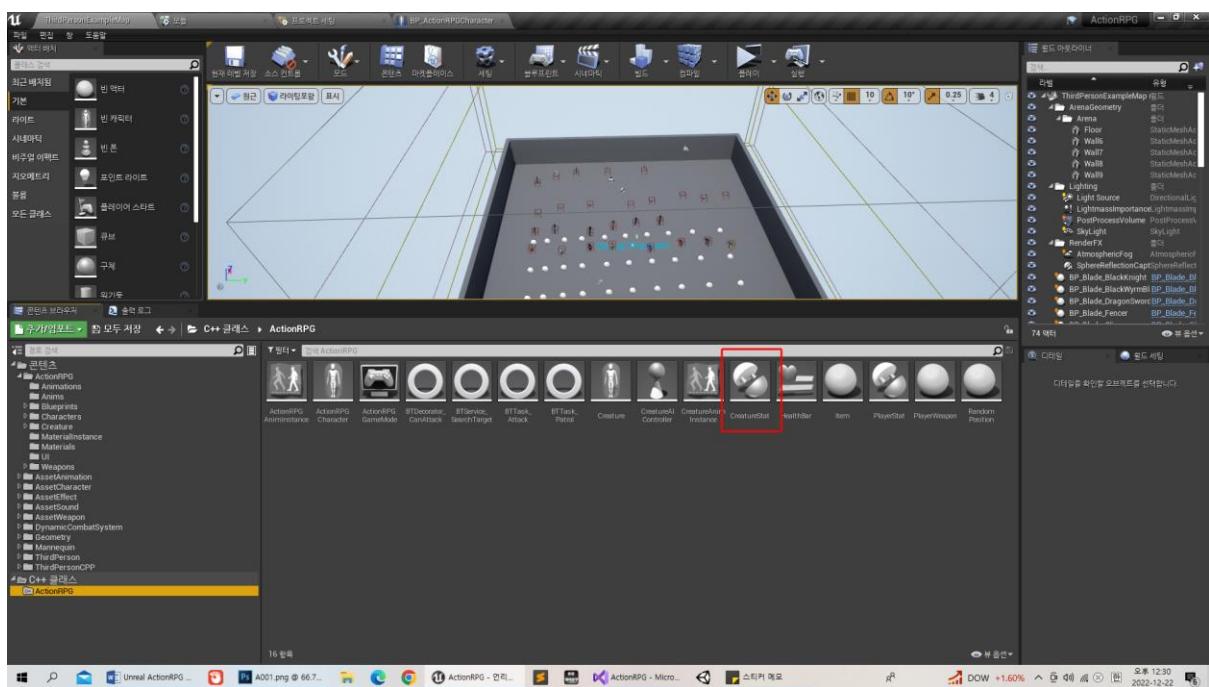
Screenshot of the Microsoft Visual Studio Code editor showing the ActionRPG project. The code editor displays the Creature.cpp file with the following code:

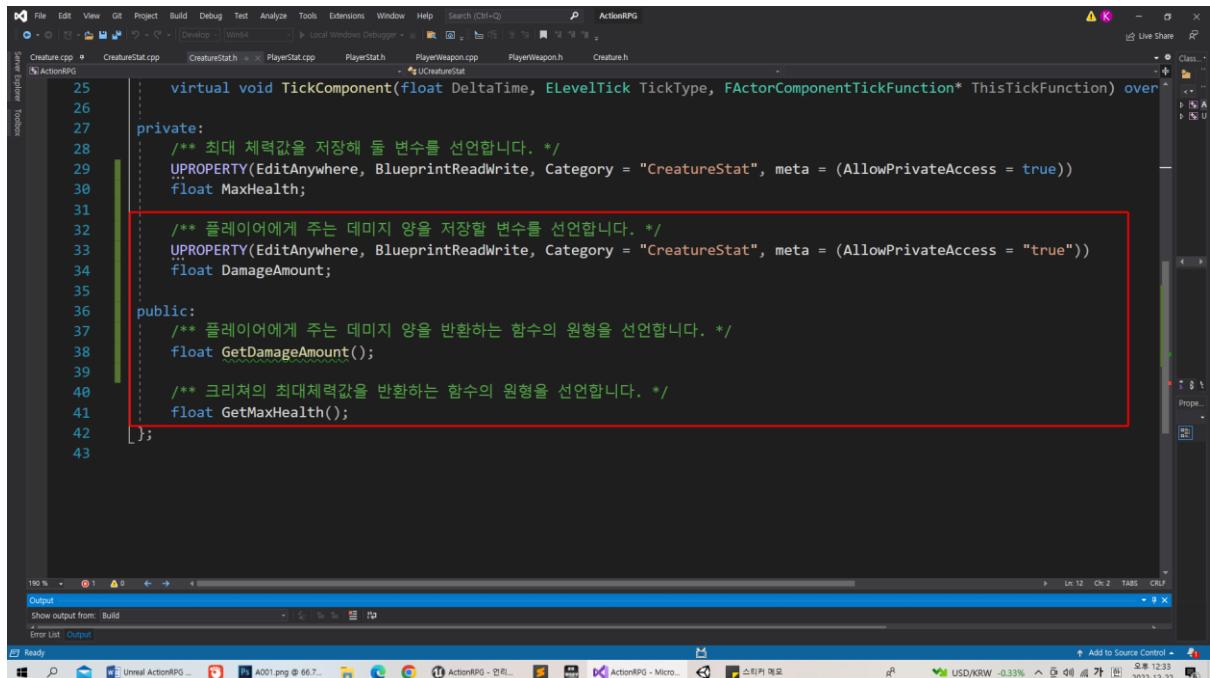
```

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 액터가 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 5초 동안 브더러게 회전하면서 히저간 */
        }

```

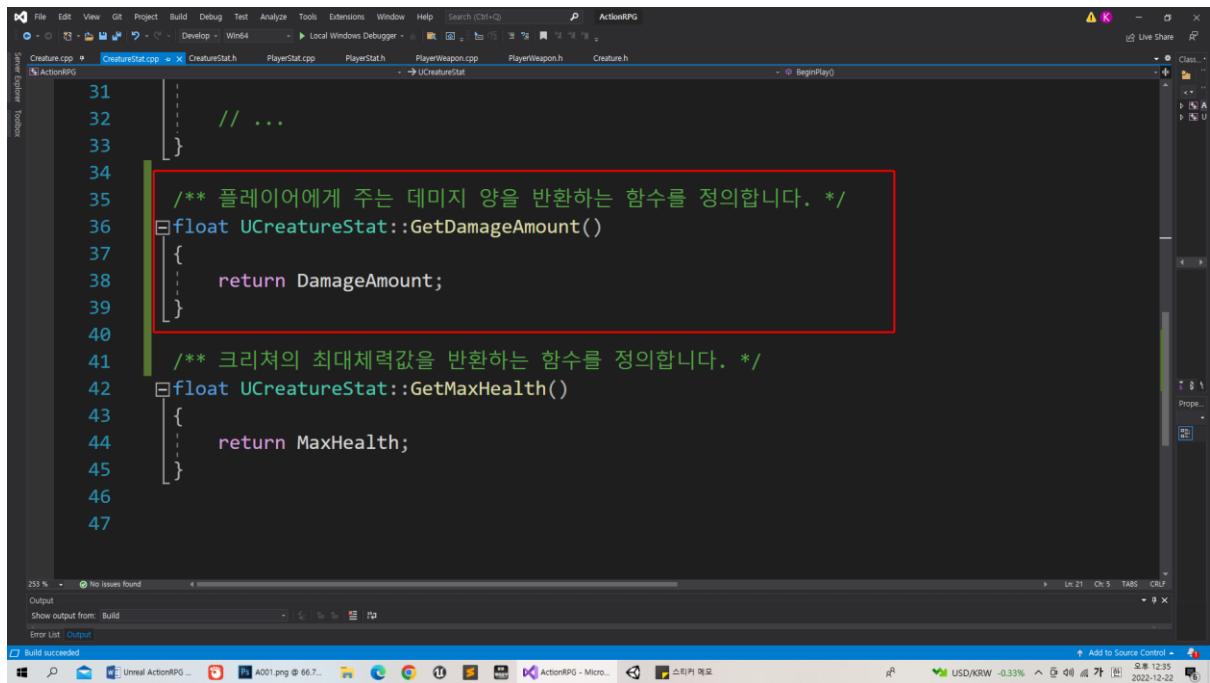
The code editor shows a red box highlighting the line `UGameplayStatics::ApplyDamage(OtherActor, 100.0f, nullptr, this, UDamageType::StaticClass());`. The status bar at the bottom right shows the date as 2022-12-22.





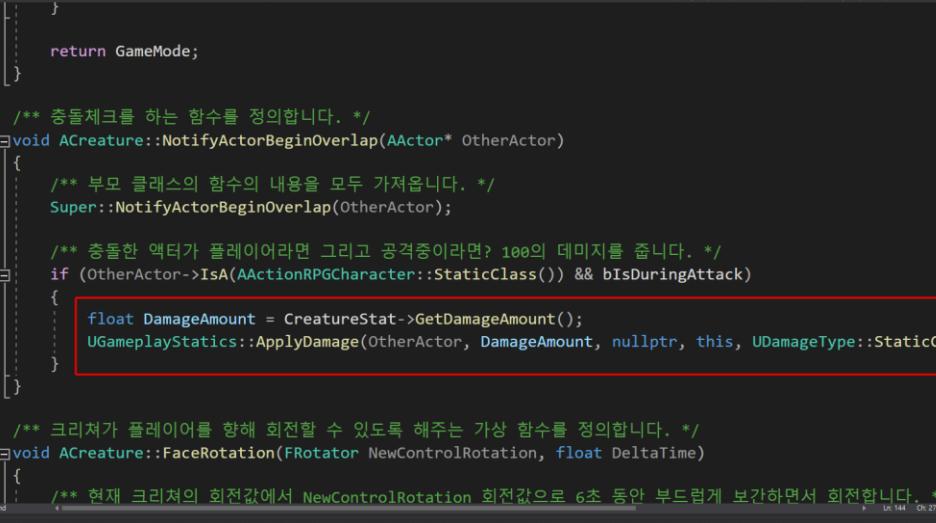
```
25     virtual void TickComponent(float DeltaTime, ELevelTick TickType, FActorComponentTickFunction* ThisTickFunction) override
26     {
27     }
28
29     private:
30         /** 최대 체력값을 저장해 둘 변수를 선언합니다. */
31         UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "CreatureStat", meta = (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.

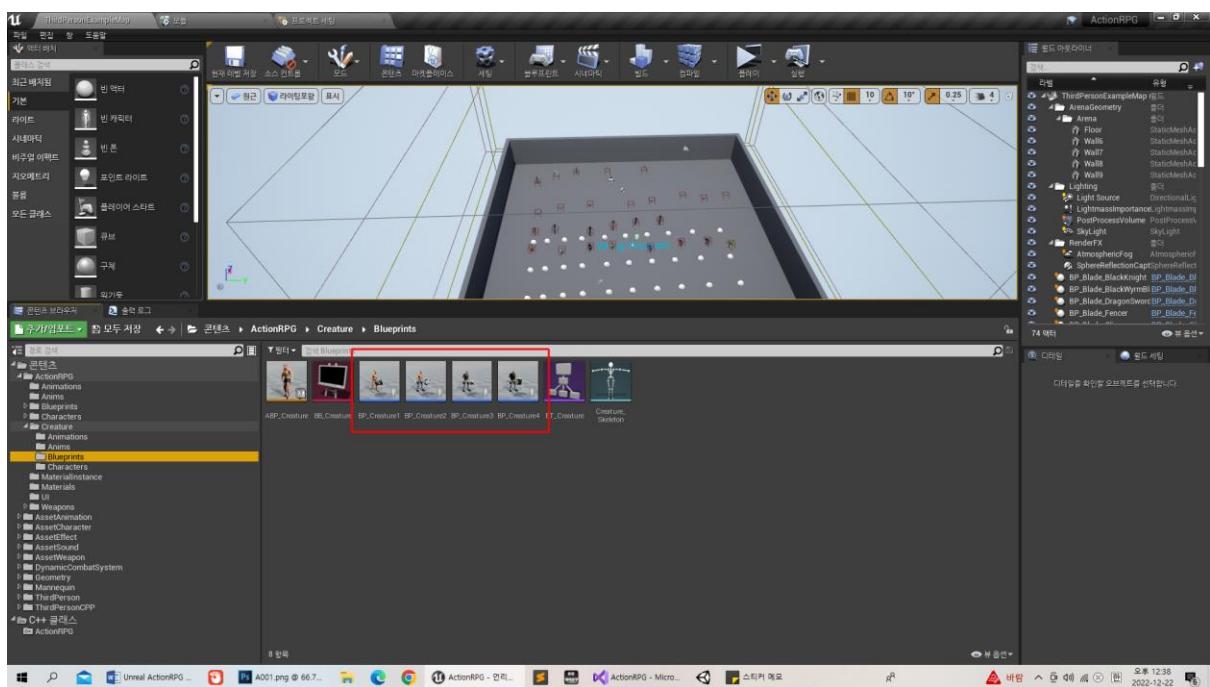


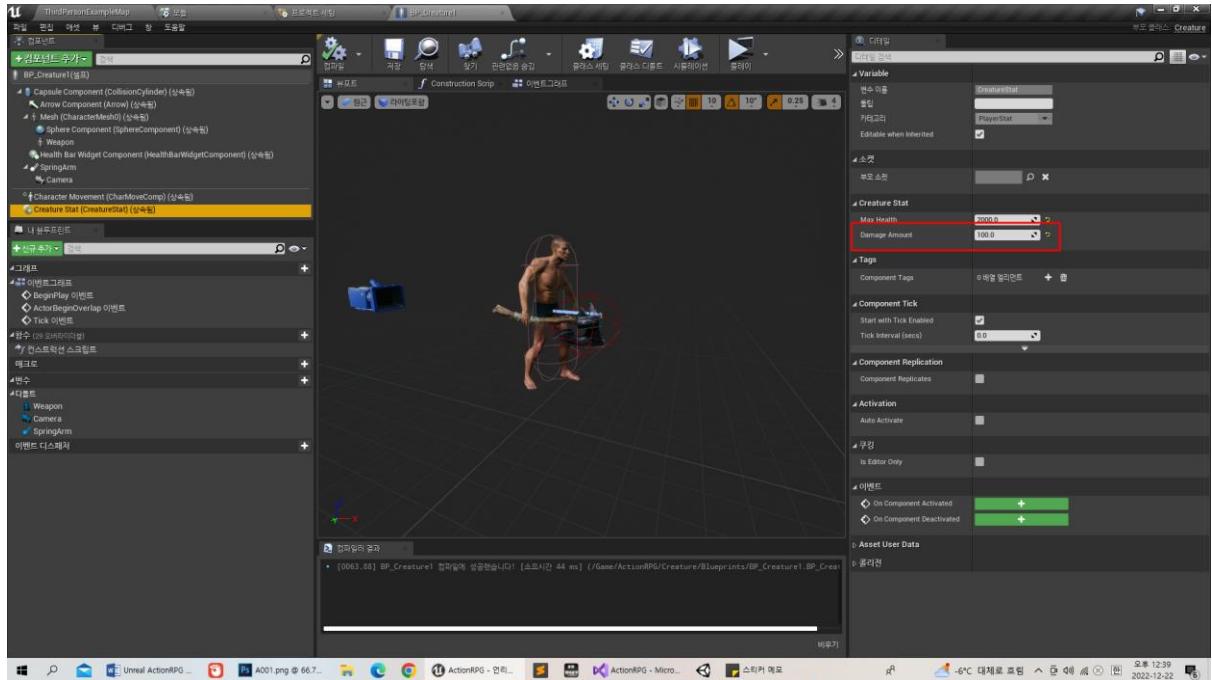
```
31
32
33
34
35     /** 플레이어에게 주는 데미지 양을 반환하는 함수를 정의합니다. */
36     float UCreatureStat::GetDamageAmount()
37     {
38         return DamageAmount;
39     }
40
41     /** 크리쳐의 최대체력값을 반환하는 함수를 정의합니다. */
42     float UCreatureStat::GetMaxHealth()
43     {
44         return MaxHealth;
45     }
46
47 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

크리쳐 클래스에서 수정해 주도록 합니다.



```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) ActionRPG
Creature.cpp CreatureStat.cpp CreatureStat.h PlayerStat.cpp PlayerStat.h PlayerWeapon.cpp PlayerWeapon.h Creature.h
ActionRPG
162     }
163
164     return GameMode;
165 }
166
167 /**
168  * 충돌체크를 하는 함수를 정의합니다. */
169 void ACreature::NotifyActorBeginOverlap(AActor* OtherActor)
170 {
171     /**
172      * 부모 클래스의 함수의 내용을 모두 가져옵니다. */
173     Super::NotifyActorBeginOverlap(OtherActor);
174
175     /**
176      * 충돌한 액터가 플레이어라면 그리고 공격중이라면? 100의 데미지를 줍니다. */
177     if (OtherActor->IsA(AActionRPGCharacter::StaticClass()) && bIsDuringAttack)
178     {
179         float DamageAmount = CreatureStat->GetDamageAmount();
180         UGameplayStatics::ApplyDamage(OtherActor, DamageAmount, nullptr, this, UDamageType::StaticClass());
181     }
182
183     /**
184      * 크리쳐가 플레이어를 향해 회전할 수 있도록 해주는 가상 함수를 정의합니다. */
185 void ACreature::FaceRotation(FRotator NewControlRotation, float DeltaTime)
186 {
187     /**
188      * 현재 크리쳐의 회전값에서 NewControlRotation 회전값으로 6초 동안 부드럽게 보간하면서 회전합니다. */
189 }
```





플레이를 해서 결과를 확인합니다.

