The UE4 to STD Cheatsheet

|  |  |  |
| --- | --- | --- |
| UE4 | STD | Notes |
| PRINTING TEXT | | |
| UE\_LOG(category, verbosity, message, …) | puts(message)  printf(message, …)  std::cout |  |
|  | puts() | #include <stdio.h>  Simply dumps text straight to a console |
|  | printf(message, …) | #include <stdio.h>  Dumps text to the console with formatting and arguments |
|  | std::cout | #include <iostream> // contains stdio.h  Cout = character output  Streams text to the console |
| NUMBERIC TYPES | | |
| int8  int16  int32  int64  uint8  uint16  uint32  uint64 | int8\_t  int16\_t  int32\_t  int64\_t  uint8\_t  uint16\_t  uint32\_t  uint64\_t | #inlcude <stdint.h>  Some int sizes (like “long” and “long long” between Windows and Linux) are uniformly sized between platforms. The stdint header fixes this problem |
| STRING OBJECT | | |
| TCHAR\* | char\*  wchar\_t\* | Pointers to blocks of memory that are of type char or wchar\_t |
| FString | std::string  std::wstring | #include <iostream>  #include <string> // special string functionality provided here |
| FName | NO EQUIVALENTS | NO EQUIVALENTS |
| ARRAY OBJECT | | |
| TArray<T> | std::vector<T> | #include <vector> |
| TArray<T>::Num() | std::vector<T>::size() |  |
| TArray<T>::Add(T) | std::vector<T>::push\_back(T) | Adds an element on at the end of the array |
| NO EQUIVALENT | std::vector<T>::pop\_back() | Removes the element at the end of the array |
| TArray<T>::RemoveAt(index) | std::vector<T>::erase(iterator) | You cannot delete by array index alone with std::vector. The next best thing to do is call begin and add the index number. So:  Int32\_t index = 4;  std::vector<int32\_t> vec;  // populate with data  vec.erase(vec.begin() + index); |
| MAPS | | |
| TMAP | std::map |  |
|  |  |  |
|  |  |  |