NGN_DISK.H

int32_t ReadBinaryFile(
**   std::string filepath,**
**   std::vector<uint8_t> &buffer**
);

Abre y lee en modo binario el archivo especificado en la ruta del sistema de archivos del sistema y lo almacena en el buffer dado. Este método, además, devuelve el tamaño del archivo leído (en bytes) o -1 en caso de error.

```
std::vector<uint8_t> data;
int32_t length = ngn->disk->ReadBinaryFile(“data/gamelevel.bin”, data);
```

int32_t WriteBinaryFile(
**   std::string filepath,**
**   std::vector<uint8_t> &buffer**
);

Abre y escribe en modo binario, en el archivo especificado en la ruta del sistema de archivos del sistema, los datos almacenados en el buffer dado. Este método, además, crea la ruta si esta no existe y devuelve el número de bytes escritos en el archivo o -1 en caso de error.

```
std::vector<uint8_t> save_data;
int32_t length = ngn->disk->WriteBinaryFile(“save/card01.sav”, save_data);
```

// Primera sobrecarga;
std::string ReadTextFile(std::string filepath);

// Segunda sobrecarga;
bool ReadTextFile(
**   std::string filepath,**
**   std::vector<std::string> &lines**
);

Abre y lee en modo texto el archivo especificado en la ruta del sistema de archivos del sistema y devuelve su contenido en un string o una cadena vacia si no se puede abrir (primera sobrecarga). La segunda sobrecarga almacena por separado en el vector de strings dado las líneas de texto, además de devolver TRUE o FALSE según si se ha podido o no leer el archivo.

```
// Primera sobrecarga
std::string text = ngn->disk-> ReadTextFile(“data/info/eula.txt”);

// Segunda sobrecarga
std::vector<std::string> text_lines;
bool r = ngn->disk->ReadTextFile(“data/info/eula.txt”, text_lines);
```

int32_t WriteTextFile(
**   std::string filepath,**
**   std::string text,**
**   bool append = false**
);

Abre y escribe en modo texto, en el archivo especificado en la ruta del sistema de archivos del sistema, los datos almacenados en el string dado. Este método, además, crea la ruta si esta no existe y devuelve el número de caracteres escritos (saltos de línea incluidos) en el archivo o -1 en caso de error. Opcionalmente, se puede especificar si los datos han de ser adjuntados al final del archivo, conservando los existentes (append = true) o se ha de sobrescribir el contenido del archivo (append = false, por defecto).

```
std::string txt = “This is a simple text”;
int32_t length = ngn->disk->WriteTextFile(“logs/debug.log”, txt, true);
```

int32_t CheckFile(std::string path);

Verifica si el archivo especificado en la ruta existe y es accesible. En caso afirmativo, devuelve el tamaño de dicho archivo en bytes. De no serlo, devuelve -1.

```
int32_t length = ngn->disk->CheckFile(“logs/debug.log”);
```