GrdHash

Функция(метод) **GrdHash** вычисляет хэш-функцию блока данных.

C

```
int GRD_API GrdHash(
  HANDLE hGrd,
  DWORD dwHash,
  DWORD dwDataLng,
  void *pData,
  DWORD dwMethod,
  void *pDigest,
  void *pKeyBuf,
  void *pContext
);
```

hGrd	хэндл, через который будет выполнена данная операция
dwHash	Номер аппаратно- или программно-реализованного алгоритма, вычисляющего хэш-функцию. Номер программно-реализованного алгоритма должен быть должен быть равен **GrdSH_CRC32** или **GrdSH_SHA256.** Номер аппаратного алгоритма должен соответствовать номеру используемого алгоритма **HASH64.** GrdSH_CRC32 — Алгоритм CRC32 GrdSH_SHA256 — Алгоритм SHA256
dwDataLng	Длина блока данных, хэш которых будет вычисляться, в байтах. Для алгоритмов **HASH64** минимальная длина блока составляет **GrdARS_HASH64.**
pData	указатель на буфер данных , хэш которых будет вычисляться
dwMethod	метод преобразования, который задается комбинацией флагов **GrdSC_XXX** GrdSC_First — Первый блок данных GrdSC_Next — Следующий блок данных GrdSC_Last — Последний блок данных GrdSC_All — Единственный блок данных. Хэш считается за один раз
pDigest	Указатель на буфер, куда будет помещен результат вычислений. Для этого буфера должна быть зарезервирована память не менее: • для алгоритма **SHA256** - **GrdSHA256_DIGEST_SIZE** • для алгоритма **HASH64** - **GrdHASH64_DIGEST_SIZE** • для алгоритма **CRC32** - **GrdCRC32_DIGEST_SIZE**
pKeyBuf	зарезервировано. Параметр должен быть равен **NULL**
pContext	Буфер, длиной **GrdSHA256_CONTEXT_SIZE** для хранения контекста (состояния алгоритма) при последовательном вычислении хэш-функции **SHA256** от нескольких блоков данных. Для алгоритмов **HASH64** и **CRC32** должен быть равен **NULL**. GrdSHA256_DIGEST_SIZE — Размер дайджеста, возвращаемого алгоритмом **SHA256** GrdSHA256_CONTEXT_SIZE — Размер контекста алгоритма **SHA256** для запоминания предыдущего состояния алгоритма

Набор ошибок Guardant API

Функция **GrdHash** вычисляет хэш-функцию блока данных *pData* длиной *dwDataLng*.

В Guardant API реализовано вычисление программно-реализованных функций **CRC32** и **SHA256**, а также аппаратных алгоритмов **HASH64**. Выбор алгоритма осуществляется параметром *dwHash*, который может принимать значения констант **GrdSH_CRC32** и **GrdSH_SHA256** для программно-реализованных алгоритмов, либо номера используемого аппаратного алгоритма **HASH64**.

Все хэш-функции могут вычисляться от больших блоков данных, поэтому предусмотрена возможность разбиения данных на меньшие блоки и вычисления значения хэша последовательно для нескольких буферов. Для этого параметром *dwMethod* задается порядок блока **GrdSC_XXX** (первый, следующий, последний).

Для передачи состояния алгоритма **SHA256** используется специальный контекст *pContext*, память для которого должна быть зарезервирована и проинициализирована заранее. Для алгоритмов **HASH64** и **CRC32** контекст не используется. Функция **GrdHash** самостоятельно разбивает буфер на блоки необходимой длины и выполняет все операции по согласованию.

Если операция выполняется за один прием, то параметром *dwMethod* должен быть задан метод **GrdSC_All**.

При работе с аппаратным алгоритмом **HASH64** рекомендуется использовать блок данных *pData*, длина которого *dwDataLng* кратна 16 байтам (**GrdARS_HASH64**). Если длина блока данных не будет кратной 16, то буфер дополнится нулями до ближайшего значения, кратного 16. Поэтому для воспроизведения итогового значения хеша при "некратных" значениях размера буфера, важны не только последовательности их обработки, но и размеры *dwDataLng*.

На время вычисления **HASH64** блокируется выполнение любых алгоритмов на данном ключе. Поэтому передача буфера большого размера может надолго заблокировать электронный ключ.

Результат вычислений помещается в буфер *pDigest*, память для которого размером, соответствующим конкретному алгоритму, должна быть зарезервирована заранее.

Примечание

> Результат вычислений возвращается в память по адресу *pDigest* только в момент вызова с флагом **GrdSC_Last** (или **GrdSC_All).** До этого промежуточный результат хранится во внутреннем буфере ключа, вследствие чего (с целью его сохранности) при вычислении hash от длинных блоков данных в несколько подходов рекомендуется блокировать обращение к этой функции путем вызова GrdLock с соответствующими параметрами.

C#

```
public static GrdE GrdHash(Handle grdHandle, GrdAlgNum hashNum, byte[] data, GrdSC method, byte[] digest)
public static GrdE GrdHash(Handle grdHandle, GrdAlgNum hashNum, byte[] data, GrdSC method, byte[] digest, byte
[] context)
```

grdHandle [in]

Тип: Handle

хэндл, через который будет выполнена данная операция.

hashNum [in]

Тип: GrdAlgNum

Номер аппаратно- или программно-реализованного алгоритма, с помощью которого будет вычисляться хэш-функция.

data [in]

Тип: byte [ ]

Указатель на буфер данных, хэш которых будет вычисляться.

method [in]

Тип: GrdSC

Метод преобразования, который задается комбинацией флагов GrdSC.

digest [in]

Тип: byte [ ]

Указатель на буфер, куда будет помещен результат вычислений.

context [in]

Тип: byte [ ]

Буфер для хранения контекста (состояния алгоритма) при последовательном вычислении хэш функции SHA256 от нескольких блоков данных.
Набор ошибок Guardant API
Метод **GrdHash** вычисляет хэш-функцию блока данных *data*.

В Guardant API реализовано вычисление программно-реализованных функций **CRC32** и **SHA256**, а также аппаратных алгоритмов **HASH64**. Выбор алгоритма осуществляется параметром *hashNum*, который может принимать значения констант GrdSA.CRC32 и GrdSA.SHA256 для программно-реализованных алгоритмов, либо номера используемого аппаратного алгоритма **HASH64**.

Все хэш-функции могут вычисляться от больших блоков данных, поэтому предусмотрена возможность разбиения данных на меньшие блоки и вычисления значения хэша последовательно для нескольких буферов. Для этого параметром *method* задается порядок блока GrdSC.XXX (первый, следующий, последний).

Для передачи состояния алгоритма **SHA256** используется специальный контекст *context*, память для которого должна быть зарезервирована и проинициализирована заранее. Для алгоритмов **HASH64** и **CRC32** контекст не используется. Метод **GrdHash** самостоятельно разбивает буфер на блоки необходимой длины и выполняет все операции по согласованию.

Если операция выполняется за один прием, то параметром *method* должен быть задан метод GrdSC.All.

При работе с аппаратным алгоритмом **HASH64** рекомендуется использовать блок данных *data*, длина которого кратна 16 байтам (GrdARS.HASH64). Если длина блока данных не будет кратной 16, то буфер дополнится нулями до ближайшего значения, кратного 16. Поэтому для воспроизведения итогового значения хеша при "некратных" значениях размера буфера, важны не только последовательности их обработки, но и их размеры.

На время вычисления **HASH64** блокируется выполнение любых алгоритмов на данном ключе. Поэтому передача буфера большого размера может надолго заблокировать электронный ключ.

Результат вычислений помещается в буфер *digest*, память для которого размером, соответствующим конкретному алгоритму, должна быть зарезервирована заранее.

Примечание

> Результат вычислений возвращается в память по адресу *digest* только в момент вызова с флагом GrdSC.Last (или GrdSC.All). До этого промежуточный результат хранится во внутреннем буфере ключа, вследствие чего (с целью его сохранности) при вычислении hash от длинных блоков данных в несколько подходов рекомендуется блокировать обращение к этой функции путем вызова GrdLock с соответствующими параметрами.

Java

```
public static GrdE GrdHash(Handle grdHandle, int hash,  byte[] data, GrdSC Method, byte[] digest, byte[]
context)
public static GrdE GrdHash(Handle grdHandle, int hash,  byte[] data, GrdSC Method, byte[] digest)
```

grdHandle [in]

Тип: Handle

хэндл, через который будет выполнена данная операция.

hashNum [in]

Тип: int

Номер аппаратно- или программно-реализованного алгоритма, с помощью которого будет вычисляться хэш-функция.

data [in]

Тип: byte [ ]

Указатель на буфер данных, хэш которых будет вычисляться.

Method [in]

Тип: GrdSC

Метод преобразования, который задается комбинацией флагов GrdSC.

digest [in]

Тип: byte [ ]

Указатель на буфер, куда будет помещен результат вычислений.

context [in]

Тип: byte [ ]

Буфер для хранения контекста (состояния алгоритма) при последовательном вычислении хэш функции SHA256 от нескольких блоков данных.
Набор ошибок Guardant API
Метод **GrdHash** вычисляет хэш-функцию блока данных *data*.

В Guardant API реализовано вычисление программно-реализованных функций **CRC32** и **SHA256**, а также аппаратных алгоритмов **HASH64**. Выбор алгоритма осуществляется параметром *hashNum*, который может принимать значения констант GrdSA.CRC32 и GrdSA.SHA256 для программно-реализованных алгоритмов, либо номера используемого аппаратного алгоритма **HASH64**.

Все хэш-функции могут вычисляться от больших блоков данных, поэтому предусмотрена возможность разбиения данных на меньшие блоки и вычисления значения хэша последовательно для нескольких буферов. Для этого параметром *method* задается порядок блока GrdSC.XXX (первый, следующий, последний).

Для передачи состояния алгоритма **SHA256** используется специальный контекст *context*, память для которого должна быть зарезервирована и проинициализирована заранее. Для алгоритмов **HASH64** и **CRC32** контекст не используется. Метод **GrdHash** самостоятельно разбивает буфер на блоки необходимой длины и выполняет все операции по согласованию.

Если операция выполняется за один прием, то параметром *method* должен быть задан метод GrdSC.All.

При работе с аппаратным алгоритмом **HASH64** рекомендуется использовать блок данных *data*, длина которого кратна 16 байтам (GrdARS.HASH64). Если длина блока данных не будет кратной 16, то буфер дополнится нулями до ближайшего значения, кратного 16. Поэтому для воспроизведения итогового значения хеша при "некратных" значениях размера буфера, важны не только последовательности их обработки, но и их размеры.

На время вычисления **HASH64** блокируется выполнение любых алгоритмов на данном ключе. Поэтому передача буфера большого размера может надолго заблокировать электронный ключ.

Результат вычислений помещается в буфер *digest*, память для которого размером, соответствующим конкретному алгоритму, должна быть зарезервирована заранее.

Примечание

> Результат вычислений возвращается в память по адресу *digest* только в момент вызова с флагом GrdSC.Last (или GrdSC.All). До этого промежуточный результат хранится во внутреннем буфере ключа, вследствие чего (с целью его сохранности) при вычислении hash от длинных блоков данных в несколько подходов рекомендуется блокировать обращение к этой функции путем вызова GrdLock с соответствующими параметрами.