

**SeqIO**

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> SeqIO		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		July 22, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>SeqIO</b>	<b>1</b>
1.1	SeqIO . . . . .	1
1.2	TMP:Modula-2/SeqIO.def . . . . .	1

## Chapter 1

# SeqIO

### 1.1 SeqIO

Prozeduren

CloseSeq	OpenSeqIn	OpenSeqOut
SeqGetB	SeqGetL	SeqGetLQuick
SeqGetW	SeqInB	SeqInCount
SeqInL	SeqInLQuick	SeqInLen
SeqInPos	SeqInW	SeqOk
SeqOutB	SeqOutCount	SeqOutL
SeqOutLQuick	SeqOutPos	SeqOutW

Typ-Deklarationen

SeqKey

### 1.2 TMP:Modula-2/SeqIO.def

```
DEFINITION MODULE SeqIO;
(* 7.6.90/bp *)
(*$ NameChk:=FALSE *)

FROM SYSTEM IMPORT BYTE, WORD, ADDRESS;

TYPE
  SeqKey ; (* Opaque. Geiht Di nix an! *)

(*
* Modul für gepuffertes File-I/O.
* Bei Programmende werden alle Files geschlossen.
*
* buffSize muß MINDESTENS 4 und durch 4 teilbar sein!
* Bei einem Fehler wird die Datei sofort geschlossen.
* Es ist aber trotzdem noch ein CloseSeq erforderlich.
* Es kann auch bei einem Fehler immer einfach weiter
* geschrieben werden. Beim Schreiben wird der übergebene
```

```
* Wert dann ignoriert.  
* SeqInxx liefert nach einem Fehler immer 0 zurück.  
*  
* Es werden keinerlei Überprüfungen auf einen gültigen  
* Key vorgenommen!  
* Methode beim Lesen/Schreiben:  
* OpenSeqxx, SeqIn/Outyy,  
* IF SeqOk THEN ... ELSE ... END; CloseSeqxx;  
*  
* Die ..Quick.. Prozeduren dürfen nur dann benutzt  
* werden, wenn innerhalb der gesamten Datei NUR mit  
* Langworten gearbeitet wird!  
*)
```

```
PROCEDURE OpenSeqIn(VAR key: SeqKey ; name:ARRAY OF CHAR;  
    buffSize:INTEGER):BOOLEAN;  
PROCEDURE OpenSeqOut(VAR key: SeqKey ; name:ARRAY OF CHAR;  
    buffSize:INTEGER):BOOLEAN;  
PROCEDURE CloseSeq(VAR key: SeqKey );
```

```
PROCEDURE SeqOk(key{8}: SeqKey ):BOOLEAN;  
(* TRUE, wenn soweit kein Fehler aufgetreten ist. *)
```

```
(* Die SeqGetxx rufen einfach die SeqInxx auf. *)
```

```
PROCEDURE SeqInB(key{8}: SeqKey ):CHAR;  
PROCEDURE SeqGetB(key{8}: SeqKey ; VAR b{9}:BYTE);  
PROCEDURE SeqInW(key{8}: SeqKey ):INTEGER;  
PROCEDURE SeqGetW(key{8}: SeqKey ; VAR w{9}:WORD);  
PROCEDURE SeqInL(key{8}: SeqKey ):LONGINT;  
PROCEDURE SeqGetL(key{8}: SeqKey ; VAR l{9}:ADDRESS);  
PROCEDURE SeqInLQuick(key{8}: SeqKey ):LONGINT;  
PROCEDURE SeqGetLQuick(key{8}: SeqKey ; VAR l{9}:ADDRESS);  
PROCEDURE SeqInCount(key: SeqKey ; adr:ADDRESS;  
    cnt:CARDINAL);  
PROCEDURE SeqInPos(key{8}: SeqKey ):LONGINT;  
PROCEDURE SeqInLen(key{8}: SeqKey ):LONGINT;
```

```
PROCEDURE SeqOutB(key{8}: SeqKey ; b{0}:BYTE);  
PROCEDURE SeqOutW(key{8}: SeqKey ; w{0}:WORD);  
PROCEDURE SeqOutL(key{8}: SeqKey ; l{0}:ADDRESS);  
PROCEDURE SeqOutLQuick(key{8}: SeqKey ; b{0}:ADDRESS);  
PROCEDURE SeqOutCount(key{8}: SeqKey ; adr{9}:ADDRESS;  
    cnt{0}:CARDINAL);  
PROCEDURE SeqOutPos(key{8}: SeqKey ):LONGINT;  
(* Liefert die momentane Länge des Files zurück *)
```

```
END SeqIO.
```

---