

APL64: Using SKD

Contents

 skd String Key Dictionary	2
 skd Object Actions.....	2
CLEAR	3
Count.....	4
Instances	4
 skd Instance Actions.....	6
Add	6
Actions	7
Clear	8
ContainsKey	9
ContainsKeys.....	10
Count.....	12
Create.....	12
Delete.....	13
ElementAt	14
First	16
FromArray	16
FromFileXml	19
Help	20
Keys	21
Last	22
New	22
Remove	23
Self.....	24
ToArray.....	25
ToFileXml.....	26
TryAdd.....	27
TryGetValue	28
TryGetValues.....	29

UpdateItem	29
Values.....	31

□skd String Key Dictionary

Purpose:

Supports creating and using .Net dictionary instances in APL64.

Syntax: result ← LeftArg □skd Action ActionArgs

Arguments:

Action is a case-insensitive character scalar, character vector or string scalar with text selected from among the available □skd actions. For example, «Create» and 'Create' are considered by the □skd system function as the Create action.

ActionArgs is the optional argument to the action.

LeftArg is a character scalar, character vector or string scalar identifying the action target

- '#' for the □skd object which contains all □skd instances
- InstanceName for a specific □skd instance. For example, since in the □skd object the InstanceName is a .Net string scalar, «Z» and 'Z' and 'Z' are considered by the □skd system function as the same InstanceName.
- If the value of □SkdSelf has been assigned an appropriate value, the left argument of □skd may be elided and will be defaulted to the value of □SkdSelf.

Result:

The result, if any, depends on the action called.

Remarks:

Some □skd actions have no result.

Context-sensitive syntax documentation for a specific □skd action available by prefixing the Action argument text with '?', e.g.

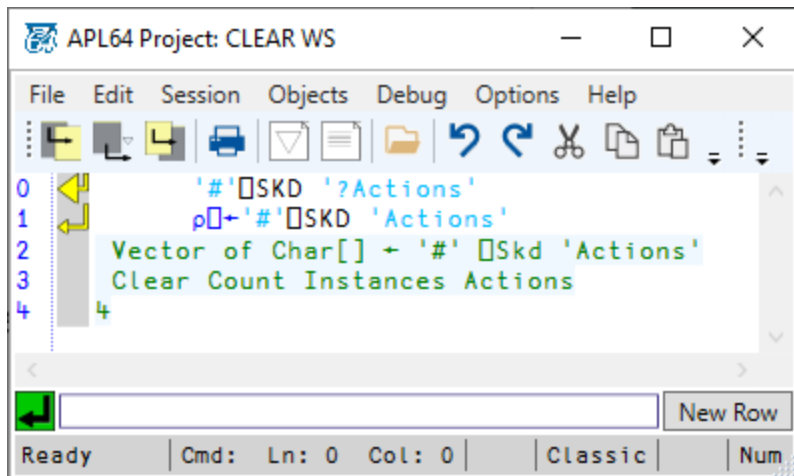
'xyz' □skd '?Clear'

APL+Win provides an ActiveX component which is similar, but not identical to □skd. It may be used in APL64 via the □wi, □cse or □na EXT interfaces.

Effect:

□skd Object Actions

The □skd object contains all □skd instances. The □skd object action names are available as a vector of character vectors:



Try It Code:

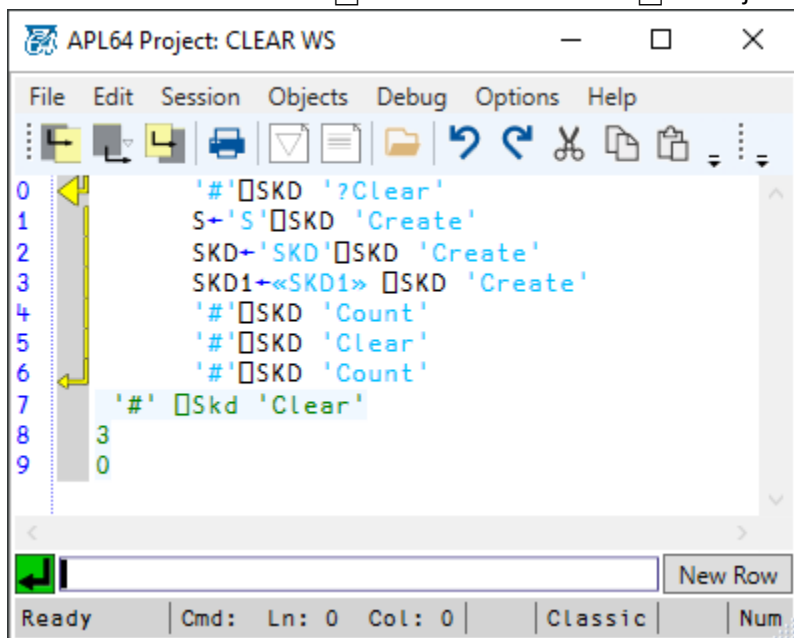
```

'#[]skd '?Actions'
p[]←'[] []skd 'Actions'

```

CLEAR

The Clear action deletes all []skd instances from the []skd object. The Clear action has no explicit result.



Try It Code:

```

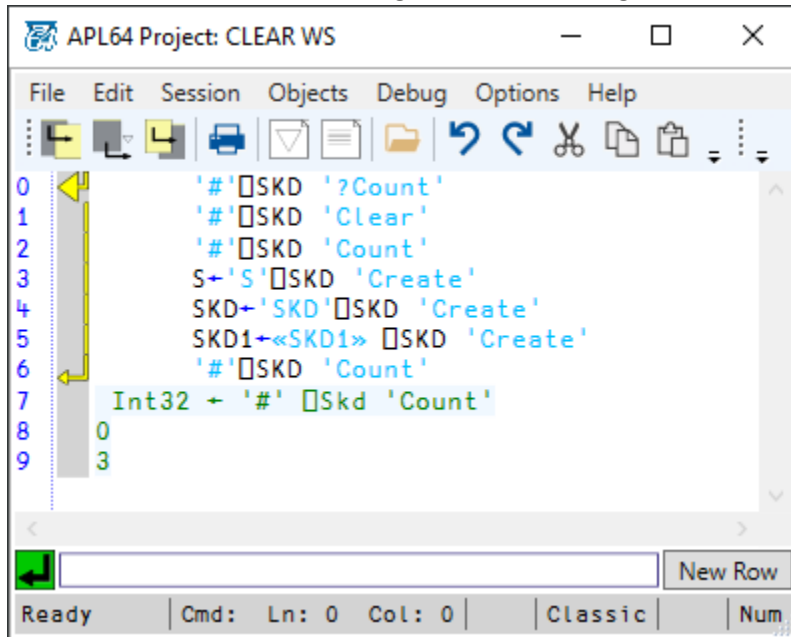
'#[]skd '?Clear'
S←'S'[]skd 'Create'
skd←'skd'[]skd 'Create'
skd1←«skd1» []skd 'Create'
'#[]skd 'Count'
'#[]skd 'Clear'

```

```
'#[]skd 'Count'
```

Count

The Count action returns an integer scalar indicating the current number of existing []skd instances.

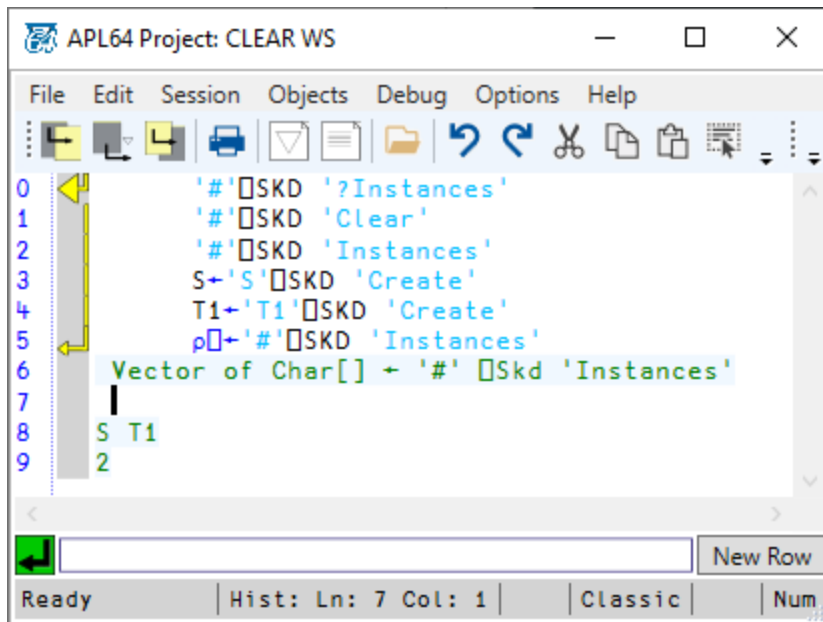


Try It Code:

```
'#[]skd '?Count'  
'#[]skd 'Clear'  
'#[]skd 'Count'  
S←'S'[]skd 'Create'  
skd←'skd'[]skd 'Create'  
skd1←«skd1» []skd 'Create'  
'#[]skd 'Count'
```

Instances

This action returns a vector of character vectors containing the names of the existing []skd instances. The left argument is the []skd system object name.



Try It Code:

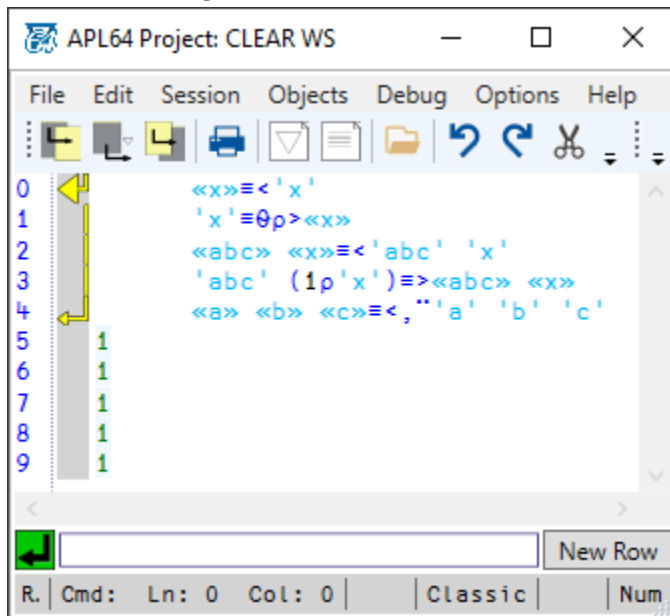
```

'#[]skd '?Instances'
'#[]skd 'Clear'
'#[]skd 'Instances'
S←'S'[]skd 'Create'
T1←'T1'[]skd 'Create'
ρ[]←'#'[]skd 'Instances'

```

□skd Instance Actions

- All □skd dictionary instances are contained in the □skd object.
- Each instance is a .Net dictionary with string scalar keys and APL object values.
- The left argument to all □skd instance actions is the name of the instance as a character scalar, character vector or string scalar, e.g.
 - «instance1» or 'instance1'
 - «X» or 'X'
- The key of an □skd dictionary <key, value> pair must be an APL64 string scalar.
- The value of an □skd dictionary <key, value> pair can be any APL64 objects.
- Conversion between character scalar, character vector and string scalars is done using the en-string and de-string functions of APL64.



Try Is Code:

```
«x»≡<'x'  
'x'≡θρ>«x»  
«abc» «x»≡<'abc' 'x'  
'abc' (1ρ'x')≡>«abc» «x»  
«a» «b» «c»≡<','a' 'b' 'c'
```

Note: The monadic > and < operators in the example are called Enstring and Destring, respectively, and are used to convert between string and character objects.

Add

This action will add the specified <key, value> pair to the specified □skd instance dictionary. The right argument of this action is the 'Add' key APLObject. The key is a character scalar, character vector or string scalar. The APLObject is the value of the specified <key, value> pair to be added to the □skd instance dictionary. An exception will be raised if the specified key exists in the □skd instance dictionary. This action has no result.

The screenshot shows the APL64 Project: CLEAR WS window. The code editor contains the following APL code:

```

0 S←'S'⊂SKD 'Create'
1 S⊂SKD '?Add'
2 S⊂SKD 'Add' «Item1» (2 3⍲6)
3 S⊂SKD 'Count'
4 S⊂SKD 'Add' «Item1» (2 3⍲6)
5 SkdInstanceName ⊂Skd 'Add' Key AplObject
6 1
7 DOMAIN ERROR: ⊂SKD: An item with the same key has already been added. Key: Item1
8 [imm:5] S⊂SKD 'Add' «Item1» (2 3⍲6)
9 ^

```

The console window at the bottom shows the error message: "DOMAIN ERROR: ⊂SKD: An item with the same key has already been added. Key: Item1". The status bar at the bottom indicates "Ready", "Cmd: Ln: 0 Col: 0", "Classic", and "Num".

Try It Code:

```

S←'S'⊂skd 'Create'
S⊂skd '?Add'
S⊂skd 'Add' «Item1» (2 3⍲6)
S⊂skd 'Count'
S⊂skd 'Add' «Item1» (2 3⍲6)

```

Actions

This action returns a vector of character vectors containing the `⊂skd` instance action names:

The screenshot shows the APL64: CLEAR WS editor window. The menu bar includes File, Edit, Session, Objects, Tools, Options, and Help. The toolbar contains various icons for file operations, editing, and navigation. The main text area displays a list of actions for a dictionary instance, starting with line 0: `S←'S'⊞skd 'Create'`. Line 1: `S⊞skd '?Actions'`. Line 2: `⇒S⊞skd 'Actions'`. Line 3: `Vector of Char[] ← SkdInstanceName ⊞Skd 'Actions'`. The subsequent lines (4-27) list actions: Add, Actions, Clear, ContainsKey, ContainsKeys, Count, Create, Delete, ElementAt, First, FromArray, FromFileXml, Keys, Last, New, Remove, Self, ToArray, ToFileXml, TryAdd, TryGetValue, TryGetValues, UpdateItem, and Values. Line 28 has a cursor. The status bar at the bottom shows 'Ready', 'Hist: Ln: 28 Col: 6 Ins', 'Classic', 'Num', and 'EN_US'.

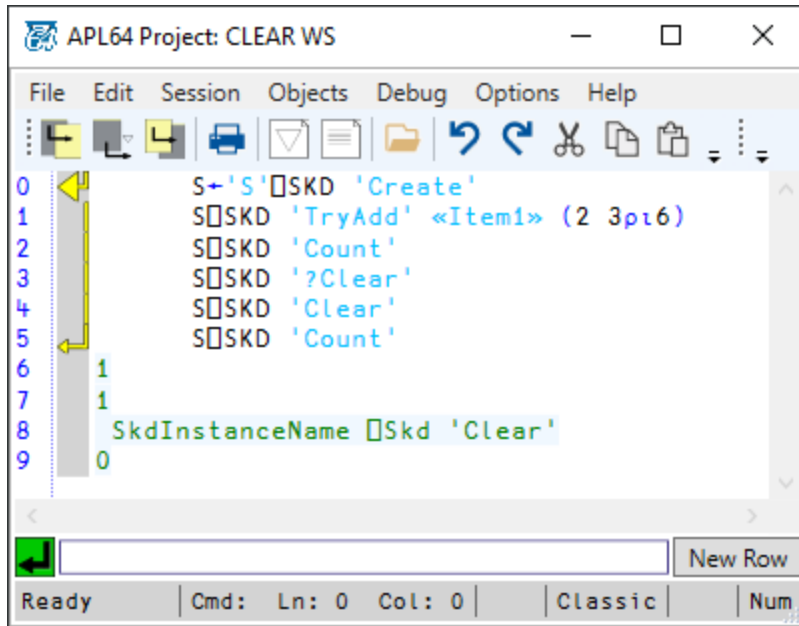
```
0      S←'S'⊞skd 'Create'
1      S⊞skd '?Actions'
2      ⇒S⊞skd 'Actions'
3      Vector of Char[] ← SkdInstanceName ⊞Skd 'Actions'
4      Add
5      Actions
6      Clear
7      ContainsKey
8      ContainsKeys
9      Count
10     Create
11     Delete
12     ElementAt
13     First
14     FromArray
15     FromFileXml
16     Keys
17     Last
18     New
19     Remove
20     Self
21     ToArray
22     ToFileXml
23     TryAdd
24     TryGetValue
25     TryGetValues
26     UpdateItem
27     Values
28     |
```

Try It Code:

```
S←'S'⊞skd 'Create'
S⊞skd '?Actions'
⇒S⊞skd 'Actions'
```

Clear

This action will remove any <key, value> pairs from the specified instance of the `⊞skd` dictionary instance specified in the left argument. This action has no result.



Try It Code:

```

S←'S'⊂skd 'Create'
S⊂skd 'TryAdd' «Item1» (2 3⍓6)
S⊂skd 'Count'
S⊂skd '?Clear'
S⊂skd 'Clear'
S⊂skd 'Count'

```

ContainsKey

This action returns a Boolean scalar indicating if the specified key is contained in the `⊂skd` dictionary instance specified in the left argument.

```

0  S←'S'⊂SKD 'Create'
1  S⊂SKD 'Add' «key1» (2 3⍲6)
2  S⊂SKD '?ContainsKey'
3  S⊂SKD 'ContainsKey' «key1»
4  S⊂SKD 'ContainsKey' «key2»
5  S⊂SKD 'ContainsKey' 'key1'
6  BoolScalar ← SkdInstanceName ⊂Skd 'ContainsKey' KeyText
7  1
8  0
9  DOMAIN ERROR: ⊂SKD: Right argument: Key: Must be string type
10 [imm:6] S⊂SKD 'ContainsKey' 'key1'
11      ^

```

Ready | Cmd: Ln: 0 Col: 0 | Classic | Num

Try It Code:

```

S←'S'⊂skd 'Create'
S⊂skd 'Add' «key1» (2 3⍲6)
S⊂skd '?ContainsKey'
S⊂skd 'ContainsKey' «key1»
S⊂skd 'ContainsKey' «key2»
S⊂skd 'ContainsKey' 'key1'

```

ContainsKeys

This action returns a Boolean vector indicating if the vector of specified keys are contained in the `⊂skd` dictionary instance specified in the left argument.

The screenshot shows the APL64 Project: CLEAR WS window. The menu bar includes File, Edit, Session, Objects, Debug, Options, and Help. The toolbar contains various icons for file operations, editing, and execution. The main workspace displays the following code and results:

```

0  S←'S'⊂SKD 'Create'
1  S⊂SKD 'Add' «abc» (2 3⍓6)
2  S⊂SKD 'Add' «a» 'abc'
3  S⊂SKD 'Add' «b» 'a'
4  S⊂SKD 'Add' «c» 'b'
5  S⊂SKD '?ContainsKeys'
6  S⊂SKD 'ContainsKeys' («abc» «a» «key3»)
7  S⊂SKD 'ContainsKeys' (1⍓«abc»)
8  S⊂SKD 'ContainsKeys' (1⍓«c»)
9  S⊂SKD 'ContainsKeys' («a» «b» «c»)
10 S⊂SKD 'ContainsKeys' (<,'a' 'b' 'c')
11 BoolVector ← SkdInstanceName ⊂Skd 'ContainsKeys' VectorOfKeys
12 1 1 0
13 1
14 1
15 1 1 1
16 1 1 1

```

The status bar at the bottom shows 'Ready', 'Cmd: Ln: 0 Col: 0', 'Classic', and 'Num'.

Try It Code:

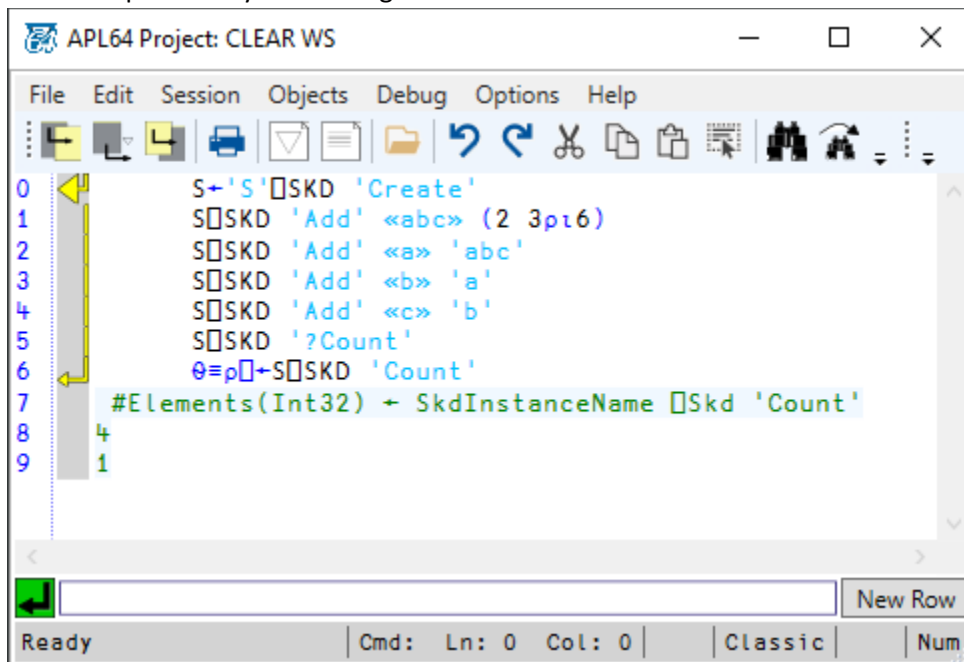
```

S←'S'⊂skd 'Create'
S⊂skd 'Add' «abc» (2 3⍓6)
S⊂skd 'Add' «a» 'abc'
S⊂skd 'Add' «b» 'a'
S⊂skd 'Add' «c» 'b'
S⊂skd '?ContainsKeys'
S⊂skd 'ContainsKeys' («abc» «a» «key3»)
S⊂skd 'ContainsKeys' (1⍓«abc»)
S⊂skd 'ContainsKeys' (1⍓«c»)
S⊂skd 'ContainsKeys' («a» «b» «c»)
S⊂skd 'ContainsKeys' (<,'a' 'b' 'c')

```

Count

This action returns the scalar integer number of <key, value> pairs in the specified `⎕skd` dictionary instance specified by the left argument.



Try It Code:

```
S←'S'⎕skd 'Create'
S⎕skd 'Add' «abc» (2 3⍓16)
S⎕skd 'Add' «a» 'abc'
S⎕skd 'Add' «b» 'a'
S⎕skd 'Add' «c» 'b'
S⎕skd '?Count'
⍊≡ρ←S⎕skd 'Count'
```

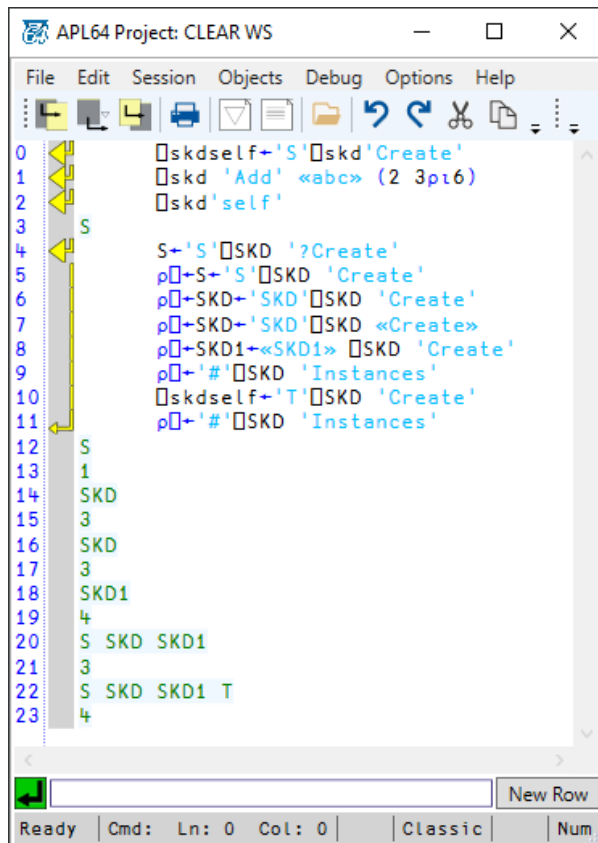
Create

This action creates a `⎕skd` string key dictionary instance in the `⎕skd` object. Any number of such instances may be created up to the memory available to APL64.

The left argument is the name of the instance. Instance names must be unique in any APL64 session.

Result is a character vector containing the text of the instance name.

If an instance with the same name exists, it will be deleted by the Create action and a new instance with the same name will be created and no exception will occur.



Try It Code:

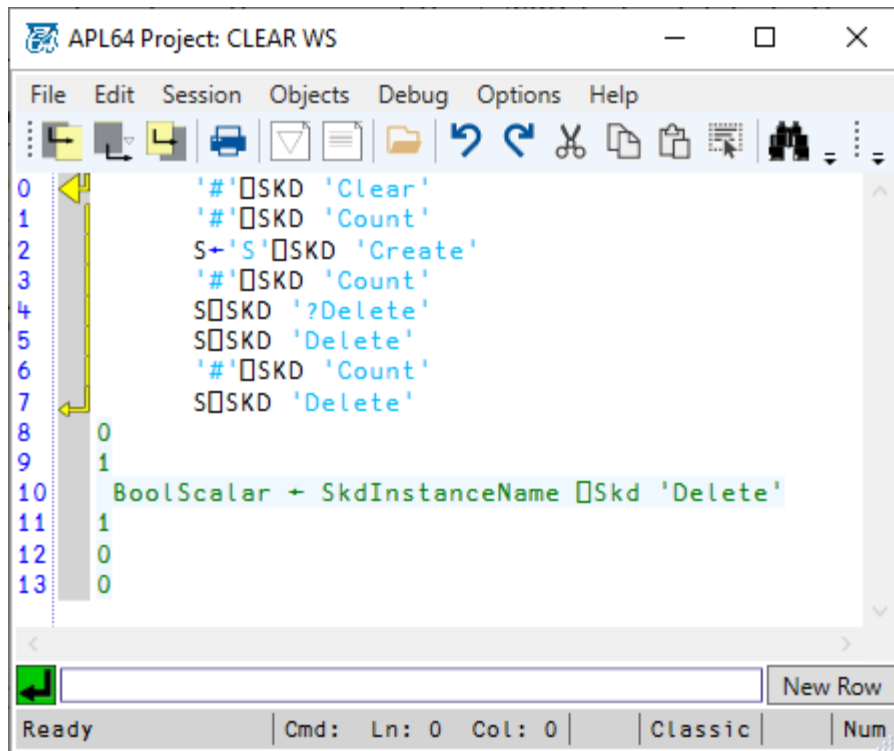
```

S←'S'⎕skd'?Create'
ρ⎕←S←'S'⎕skd'?Create'
ρ⎕←skd←'skd'⎕skd'?Create'
ρ⎕←skd←'skd'⎕skd«Create»
ρ⎕←skd1←«skd1»⎕skd'?Create'
ρ⎕←'#'⎕skd'?Instances'
⎕skdself←'T'⎕skd'?Create'
ρ⎕←'#'⎕skd'?Instances'

```

Delete

The left argument contains the name of the `⎕skd` to be deleted. This action deletes the specified `⎕skd` instance from the instances collection of the `⎕skd` object. The result is 1 if the operation succeeds, otherwise the result is 0 indicating that the instance with the specified instance does not exist.



Try It Code:

```

'# SKD 'Clear'
'# SKD 'Count'
S←'S' SKD 'Create'
'# SKD 'Count'
S SKD '?Delete'
S SKD 'Delete'
'# SKD 'Count'
S SKD 'Delete'

```

ElementAt

This action returns the <key, value> pair at the left argument integer scalar index of the SKD dictionary instance specified by the left argument. This action will throw an exception if the specified index is out of range.

```

APL64 Project: CLEAR WS
File Edit Session Objects Debug Options Help
[Icons]
0 S←'S' SKD 'Create'
1 S SKD 'Add' «abc» (2 3ρ16)
2 S SKD 'Add' «a» 'abc'
3 S SKD 'Add' «b» ('xyz' (110))
4 S SKD 'Add' «c» 'b'
5 S SKD 'Count'
6 S SKD '?ElementAt'
7 IO
8 E←E←S SKD 'ElementAt' 1
9 E≡«abc» (2 3ρ16)
10 IO←0
11 S SKD 'ElementAt' 0
12 IO←1
13 S SKD 'ElementAt' 100
14 4
15 (Key AplObject) ← SkdInstanceName SKd 'ElementAt' ElementIndex
16 1
17 abc 1 2 3
18 4 5 6
19 1
20 abc 1 2 3
21 4 5 6
22 DOMAIN ERROR: SKD: Right argument: Elt #2: Index out of range
23 [imm:14] S SKD 'ElementAt' 100
24 ^

```

Ready Cmd: Ln: 0 Col: 0 Classic Num

Try It Code:

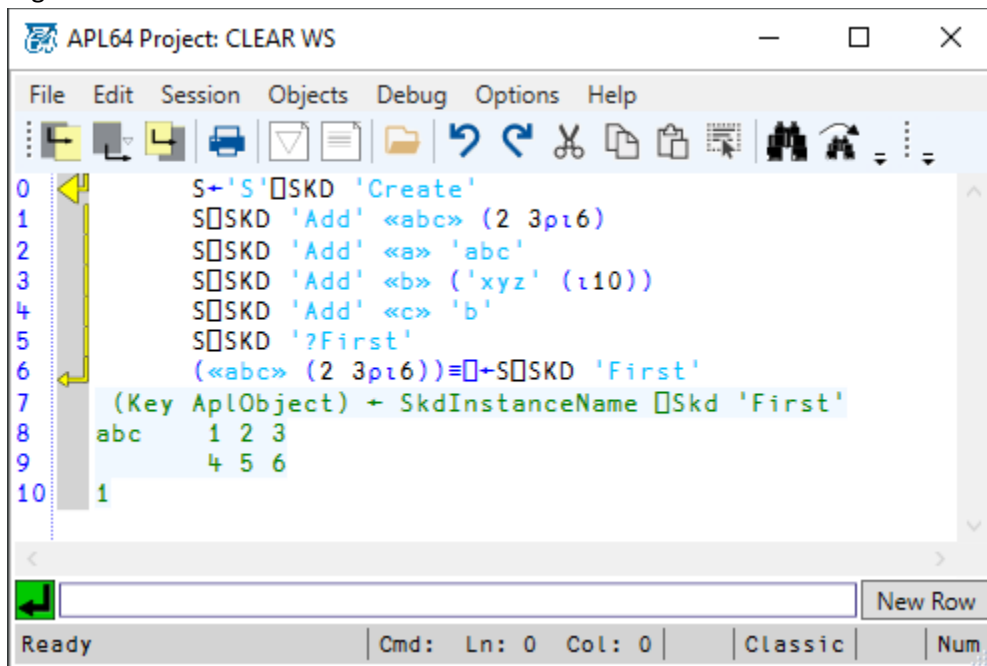
```

S←'S' skd 'Create'
S skd 'Add' «abc» (2 3ρ16)
S skd 'Add' «a» 'abc'
S skd 'Add' «b» ('xyz' (110))
S skd 'Add' «c» 'b'
S skd 'Count'
S skd '?ElementAt'
io
E←E←S skd 'ElementAt' 1
E≡«abc» (2 3ρ16)
io←0
S skd 'ElementAt' 0
io←1
S skd 'ElementAt' 100

```

First

This action returns the first <key, value> pair of the ☐skd dictionary instance specified by the left argument.



Try It Code:

```
S←'S'⊞skd 'Create'
S⊞skd 'Add' «abc» (2 3⍓16)
S⊞skd 'Add' «a» 'abc'
S⊞skd 'Add' «b» ('xyz' (110))
S⊞skd 'Add' «c» 'b'
S⊞skd '?First'
(«abc» (2 3⍓16))≡←S⊞skd 'First'
```

FromArray

This action adds the specified <key, value> pairs in the 1st right argument source data (matrix of rows of <key,value> pairs) to the target ☐skd dictionary instance specified by the left argument.

The 2nd right argument (Boolean) indicates if the target ☐skd dictionary instance should be cleared of any existing <key, value> pairs before adding the source data.

The 3rd right argument (Boolean) indicates if <key, value> pairs in the target ☐skd dictionary instance should be overwritten by <key, value> pairs in the source data if the keys are the same.

If there are duplicate keys in the source data, the last of such duplicates will be the surviving <key, value> pair to be considered to be added to the target ☐skd dictionary instance.

The FromArray action can be used to:

- Replace an `⎕skd` dictionary instance
- Merge `⎕skd` dictionary instances

```

0  S←'S'⎕SKD 'Create'
1  S⎕SKD '?FromArray'
2  'Src Data: No duplicate keys'
3  ⎕←srcData←3 2p«k1» 'v1' «k2» 'v2' «k3» 'v3 new value'
4  S⎕SKD 'FromArray' srcData 1 1
5  ⎕↑Clear the target ⎕Skd dictionary instance
6  ⎕↑Overwrite target values with source values with the same keys
7  'Result Data: '
8  ⎕←resData←S⎕SKD 'ToArray'
9  srcData≡resData
10 SkdInstanceName ⎕Skd 'FromArray' MatrixWithRows(key AplObject) Bool(ClearTargetDict) Bool(OverwriteValuesWithSameKey)
11 Src Data: No duplicate keys
12 k1 v1
13 k2 v2
14 k3 v3 new value
15 Result Data:
16 k1 v1
17 k2 v2
18 k3 v3 new value
19 1

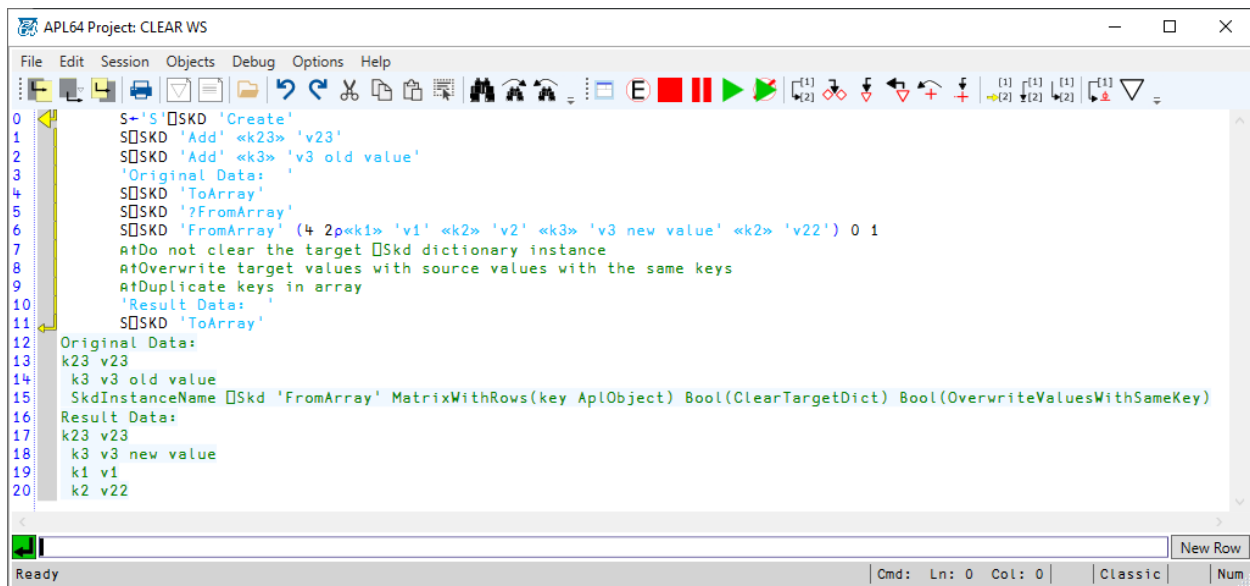
```

Try It Code:

```

S←'S'⎕skd 'Create'
S⎕skd '?FromArray'
'Src Data: No duplicate keys'
⎕←srcData←3 2p«k1» 'v1' «k2» 'v2' «k3» 'v3 new value'
S⎕skd 'FromArray' srcData 1 1
⎕↑Clear the target ⎕Skd dictionary instance
⎕↑Overwrite target values with source values with the same keys
'Result Data: '
⎕←resData←S⎕skd 'ToArray'
srcData≡resData

```

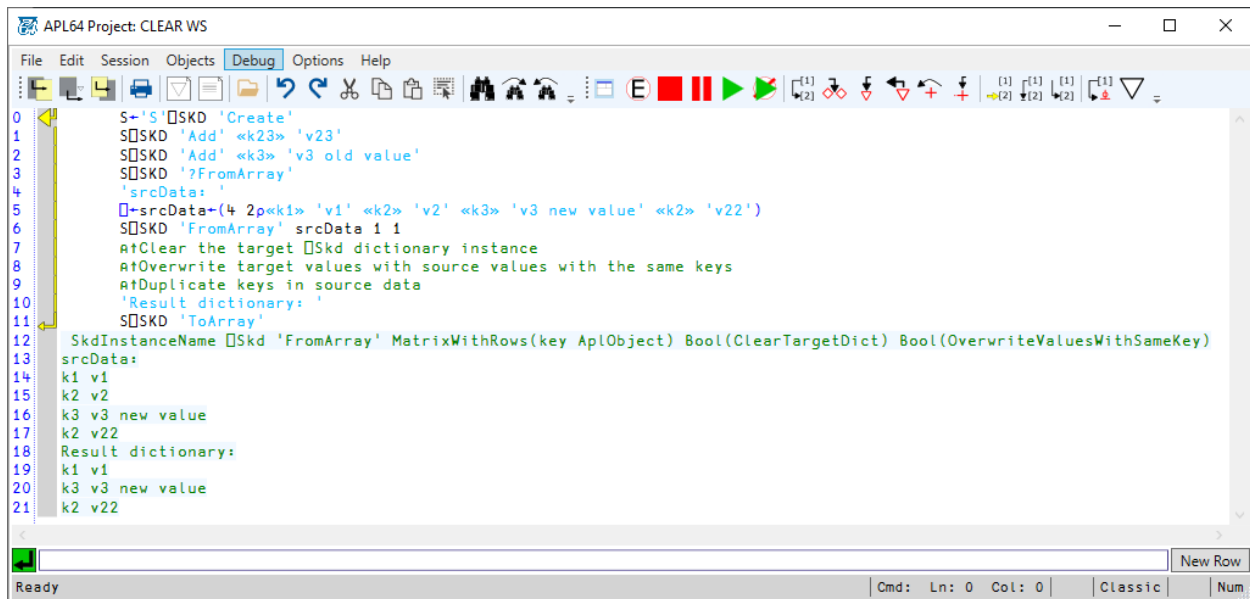


Try It Code:

```

S←'S'⊂skd 'Create'
S⊂skd 'Add' «k23» 'v23'
S⊂skd 'Add' «k3» 'v3 old value'
'Original Data: '
S⊂skd 'ToArray'
S⊂skd '?FromArray'
S⊂skd 'FromArray' (4 2p«k1» 'v1' «k2» 'v2' «k3» 'v3 new value' «k2» 'v22') 0 1
⍠Do not clear the target ⊂SKD dictionary instance
⍠Overwrite target values with source values with the same keys
⍠Duplicate keys in array
'Result Data: '
S⊂skd 'ToArray'

```



Try It Code:

```

S←'S'⎕skd 'Create'
S⎕skd 'Add' «k23» 'v23'
S⎕skd 'Add' «k3» 'v3 old value'
S⎕skd '?FromArray'
'srcData: '
⎕←srcData←(4 2p«k1» 'v1' «k2» 'v2' «k3» 'v3 new value' «k2» 'v22')
S⎕skd 'FromArray' srcData 1 1
⎕↑Clear the target ⎕SKD dictionary instance
⎕↑Overwrite target values with source values with the same keys
⎕↑Duplicate keys in source data
'Result dictionary: '
S⎕skd 'ToArray'

```

FromFileXml

This action will read the source data (file specified in the 1st right argument), xml deserialize its content and add the resulting <key, value> pairs to the target ⎕skd dictionary instance specified in the left argument.

The 2nd right argument (Boolean) indicates if the target ⎕skd dictionary instance should be cleared of any existing <key, value> pairs before adding the source data.

The 3rd right argument (Boolean) indicates if <key, value> pairs in the target ⎕skd dictionary instance should be overwritten by <key, value> pairs in the source data if the keys are the same.

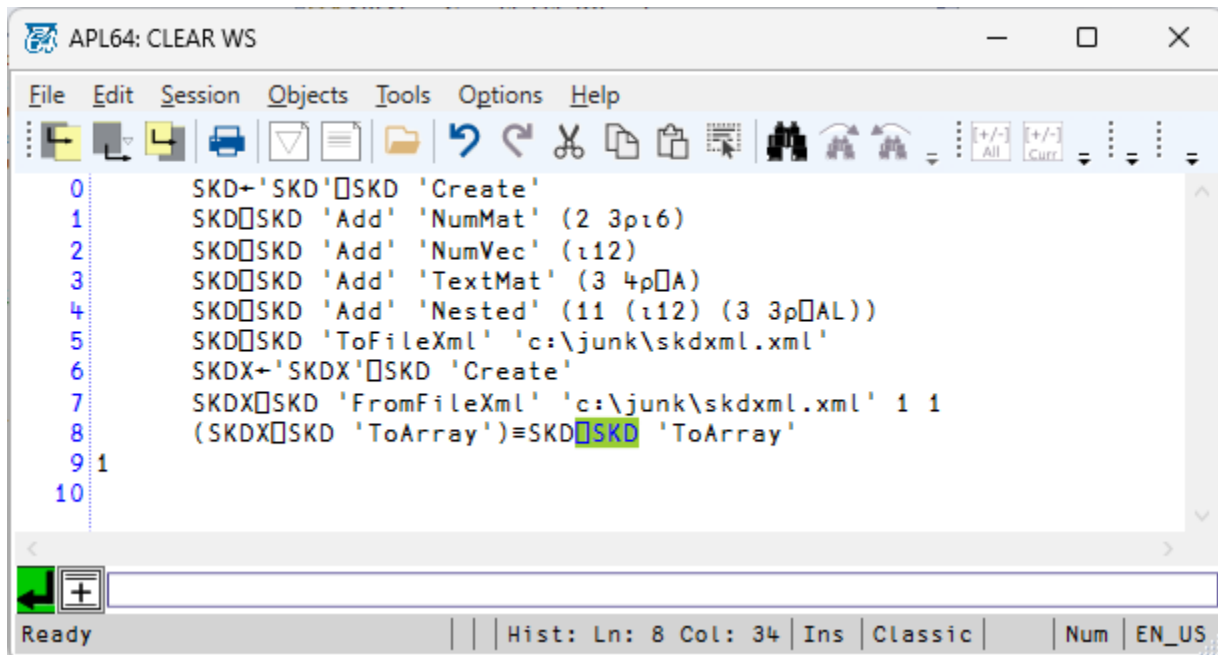
The current user must have read permission for the source data file. Modification of the serialized data will corrupt the information so that it cannot be de-serialized. This action is the inverse of the ToFileXml action.

The FromFileXml action can be used to:

- Replace an ☐skd dictionary instance
- Merge ☐skd dictionary instances
- Restore an ☐skd dictionary instance created in a previous APL64 session

Try it Code:

```
SKD←'SKD'⊖SKD 'Create'  
SKD⊖SKD 'Add' 'NumMat' (2 3p16)  
SKD⊖SKD 'Add' 'NumVec' (112)  
SKD⊖SKD 'Add' 'TextMat' (3 4p⊖A)  
SKD⊖SKD 'Add' 'Nested' (11 (112) (3 3p⊖AL))  
SKD⊖SKD 'ToFileXml' 'c:\junk\skdxml.xml'  
SKDX←'SKDX'⊖SKD 'Create'  
SKDX⊖SKD 'FromFileXml' 'c:\junk\skdxml.xml' 1 1  
(SKDX⊖SKD 'ToArray')≡SKD⊖SKD 'ToArray'
```



Help

Char[;]←☐SKD "HELP"

☐skd 'Help'

```

APL64: CLEAR WS
File Edit Session Objects Tools Options Help
0      ⌈SKD 'Help'
1      Vector of Char[] + '#' ⌈SKD 'Actions'
2      '#' ⌈SKD 'Clear'
3      Int32 + '#' ⌈SKD 'Count'
4      Vector of Char[] + '#' ⌈SKD 'Instances'
5      HelpTextSummary(char matrix) + [object or instance or null] ⌈SKD '?'
6      HelpTextSummary(char matrix) + [object or instance or null] ⌈SKD 'Help'
7      SkdInstanceName ⌈SKD 'Add' Key AplObject
8      Vector of Char[] + SkdInstanceName ⌈SKD 'Actions'
9      SkdInstanceName ⌈SKD 'Clear'
10     BoolScalar + SkdInstanceName ⌈SKD 'ContainsKey' Key
11     BoolVector + SkdInstanceName ⌈SKD 'ContainsKeys' VectorOfKeys
12     #Elements(Int32) + SkdInstanceName ⌈SKD 'Count'
13     SkdInstanceName + SkdInstanceName ⌈SKD 'Create'
14     BoolScalar + SkdInstanceName ⌈SKD 'Delete'
15     (Key AplObject) + SkdInstanceName ⌈SKD 'ElementAt' ElementIndex
16     (Key AplObject) + SkdInstanceName ⌈SKD 'First'
17     SkdInstanceName ⌈SKD 'FromArray' MatrixWithRows(key AplObject) Bool(ClearTargetDict) Bool(OverwriteValuesWithSameKey)
18     SkdInstanceName ⌈SKD 'FromFile' FileName Bool(ClearTargetDict) Bool(OverwriteValuesWithSameKey)
19     SkdInstanceName ⌈SKD 'FromFileXml' FileName Bool(ClearTargetDict) Bool(OverwriteValuesWithSameKey)
20     VectorOfKeys + SkdInstanceName ⌈SKD 'Keys'
21     (Key AplObject) + SkdInstanceName ⌈SKD 'Last'
22     SkdInstanceName + SkdInstanceName ⌈SKD 'New'
23     BoolScalar + SkdInstanceName ⌈SKD 'Remove' Key
24     SkdInstanceName + SkdInstanceName ⌈SKD 'Self'
25     MatrixOf(Key AplObject) + SkdInstanceName ⌈SKD 'ToArray'
26     SkdInstanceName ⌈SKD 'ToFile' FileName
27     SkdInstanceName ⌈SKD 'ToFileXml' FileName
28     BoolScalar + SkdInstanceName ⌈SKD 'TryAdd' (Key AplObject)
29     Vector(Bool/Success AplValue) + SkdInstanceName ⌈SKD 'TryGetValue' Key
30     MatrixOf(Bool/Success AplValue)Rows + SkdInstanceName ⌈SKD 'TryGetValues' VectorOfKeys
31     AplObject + SkdInstanceName ⌈SKD 'UpdateItem' TextKey AplObject
32     AplObject + SkdInstanceName ⌈SKD 'Values'
33
34     Notes:
35     (1) The ⌈SKD actions Add, ContainsKey, ContainsKeys, FromArray, Remove, TryAdd, TryGetValue, TryGetValues and UpdateItem
36     accept a key argument which is a text scalar, text vector or string.
37     Where applicable (i.e. multiple keys in the argument), a vector (possibly mixed) of these datatypes is accepted.
38     (2) The keys of an ⌈SKD string key dictionary are always stored and returned as strings.
39     For example, the ⌈SKD First, Keys, Last and ToArray actions provide keys as string datatypes.
40
Ready
Hist: Ln: 40 Col: 6 Ins Classic Num EN_US

```

Keys

```

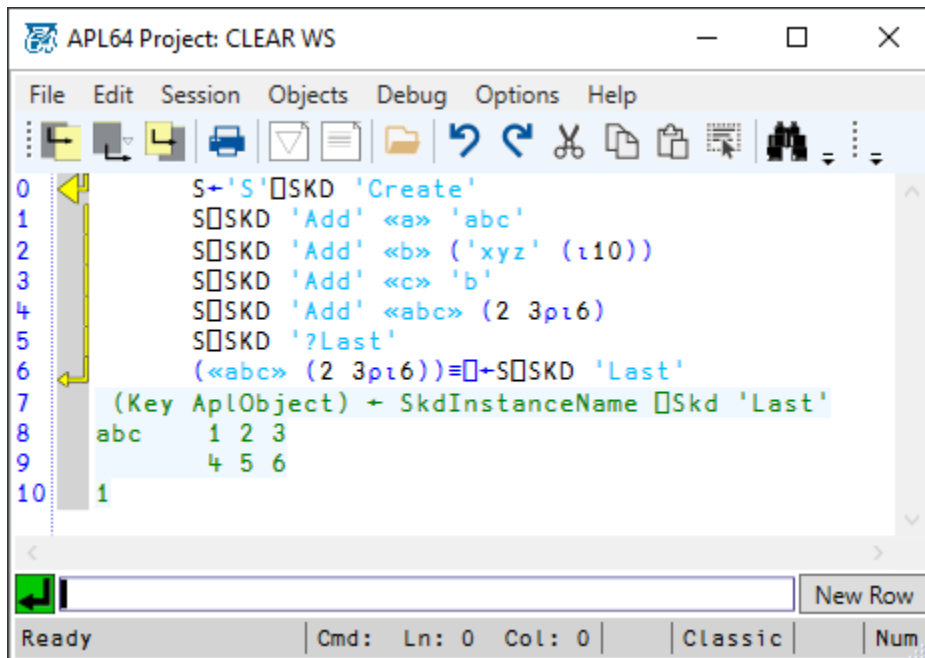
APL64 Project: CLEAR WS
File Edit Session Objects Debug Options Help
0      S←'S'⌈SKD 'Create'
1      S⌈SKD 'Add' «a» 'abc'
2      S⌈SKD 'Add' «b» ('xyz' ⌈10))
3      S⌈SKD 'Add' «c» 'b'
4      S⌈SKD 'Add' «abc» (2 3⌈16)
5      S⌈SKD '?Keys'
6      ρ⌈K←S⌈SKD 'Keys'
7      K←«a» «b» «c» «abc»
8      VectorOfKeys ← SkdInstanceName ⌈SKD 'Keys'
9      a b c abc
10     4
11     1
Ready
Cmd: Ln: 0 Col: 0 Classic Num

```

Try It Code:

```
S←'S'⊠skd 'Create'
S⊠skd 'Add' «a» 'abc'
S⊠skd 'Add' «b» ('xyz' (⊠10))
S⊠skd 'Add' «c» 'b'
S⊠skd 'Add' «abc» (2 3⊠6)
S⊠skd '?Keys'
ρ⊠←K←S⊠skd 'Keys'
K≡«a» «b» «c» «abc»
```

Last



Try It Code:

```
S←'S'⊠skd 'Create'
S⊠skd 'Add' «a» 'abc'
S⊠skd 'Add' «b» ('xyz' (⊠10))
S⊠skd 'Add' «c» 'b'
S⊠skd 'Add' «abc» (2 3⊠6)
S⊠skd '?Last'
(«abc» (2 3⊠6))≡⊠←S⊠skd 'Last'
```

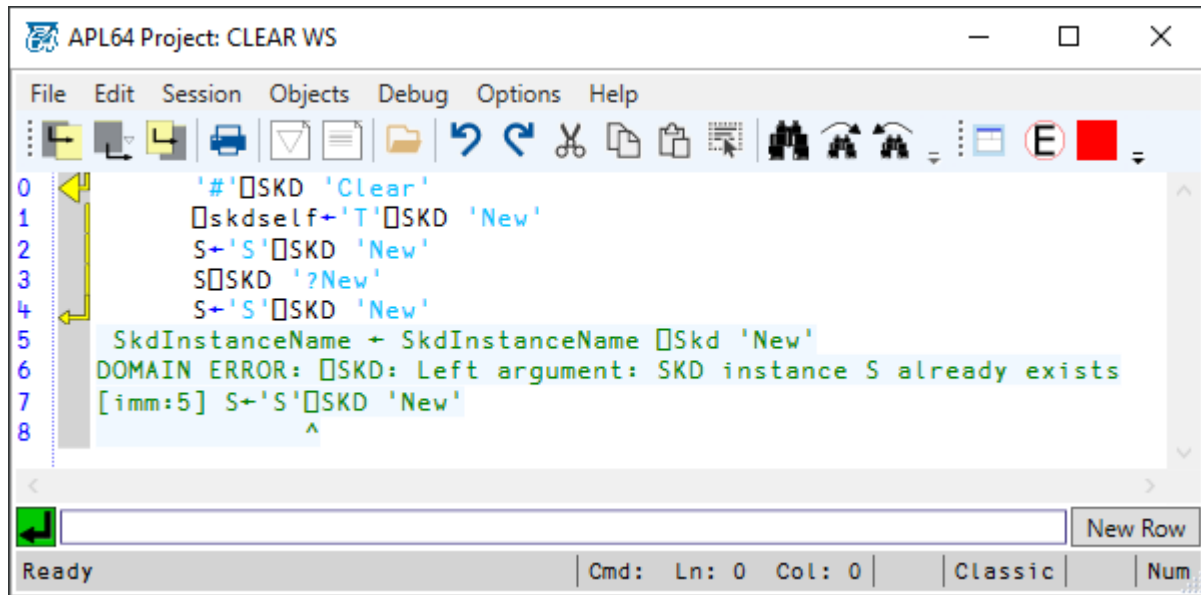
New

This action creates a `⊠skd` string key dictionary instance in the `⊠skd` object. Any number of such instances may be created up to the memory available to APL64.

The left argument is the name of the instance. Instance names must be unique in any APL64 session.

Result is a character vector containing the text of the instance name.

If an instance with the same name exists, an exception will occur, which will prevent accidental overwriting of existing `⌈skd` instances.



```
0  '#⌈SKD 'Clear'
1  ⌈skdself←'T'⌈SKD 'New'
2  S←'S'⌈SKD 'New'
3  S⌈SKD '?New'
4  S←'S'⌈SKD 'New'
5  SkdInstanceName ← SkdInstanceName ⌈Skd 'New'
6  DOMAIN ERROR: ⌈SKD: Left argument: SKD instance S already exists
7  [imm:5] S←'S'⌈SKD 'New'
8  ^
```

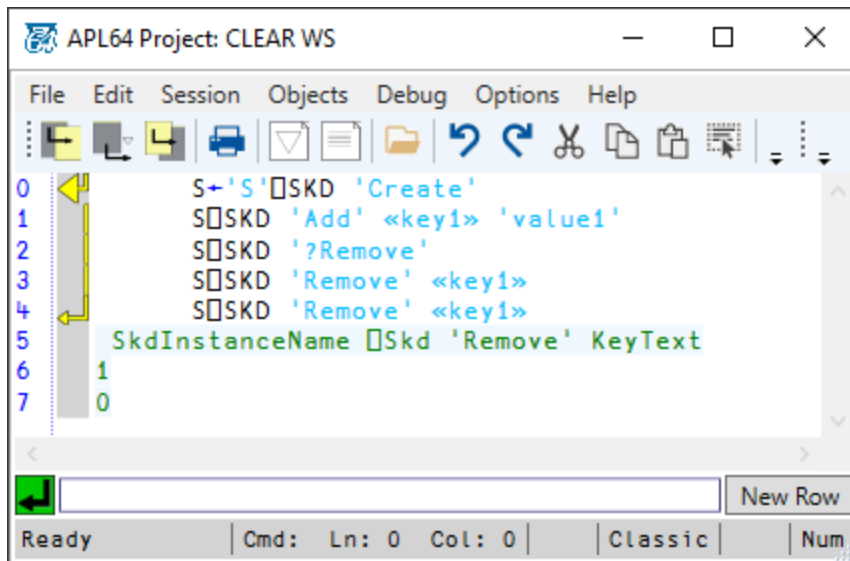
Ready | Cmd: Ln: 0 Col: 0 | Classic | Num

Try It Code:

```
'#⌈skd 'Clear'
⌈skdself←'T'⌈skd 'New'
S←'S'⌈skd 'New'
S⌈skd '?New'
S←'S'⌈skd 'New'
```

Remove

This action will remove the <key, value> pair for the specified key in the right argument in the `⌈skd` dictionary instance specified in the left argument. The Boolean scalar result indicates if the action was successful.



Try It Code:

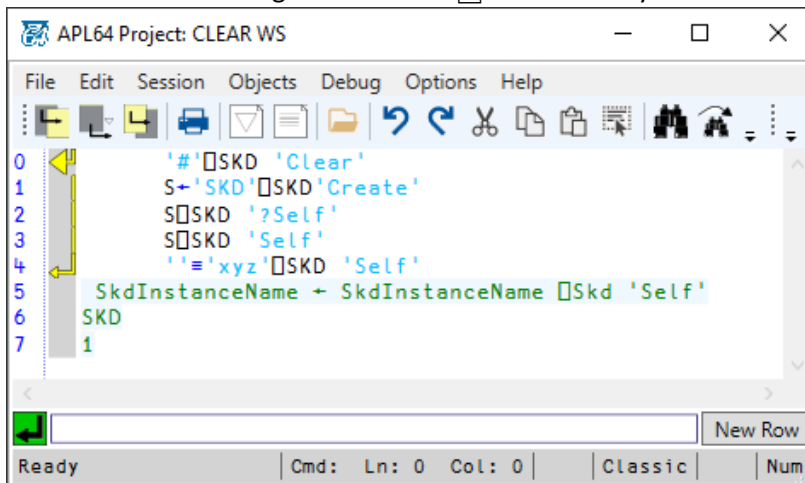
```

S←'S'⊂skd 'Create'
S⊂skd 'Add' «key1» 'value1'
S⊂skd '?Remove'
S⊂skd 'Remove' «key1»
S⊂skd 'Remove' «key1»

```

Self

This action returns a character vector containing the name of `⊂skd` dictionary instance corresponding to the value of the left argument. If the `⊂skd` dictionary instance does not exist, the result “.



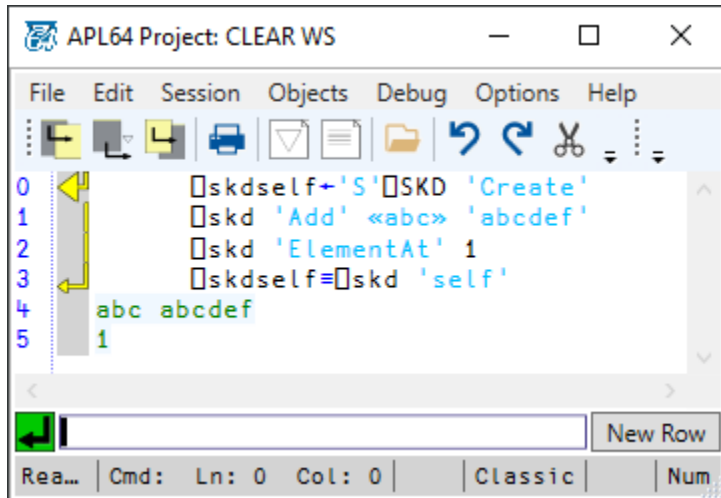
Try It Code:

```

'#⊂skd 'Clear'
S←'skd'⊂skd 'Create'
S⊂skd '?Self'
S⊂skd 'Self'

```

```
"≡'xyz'⊞skd 'Self'
```

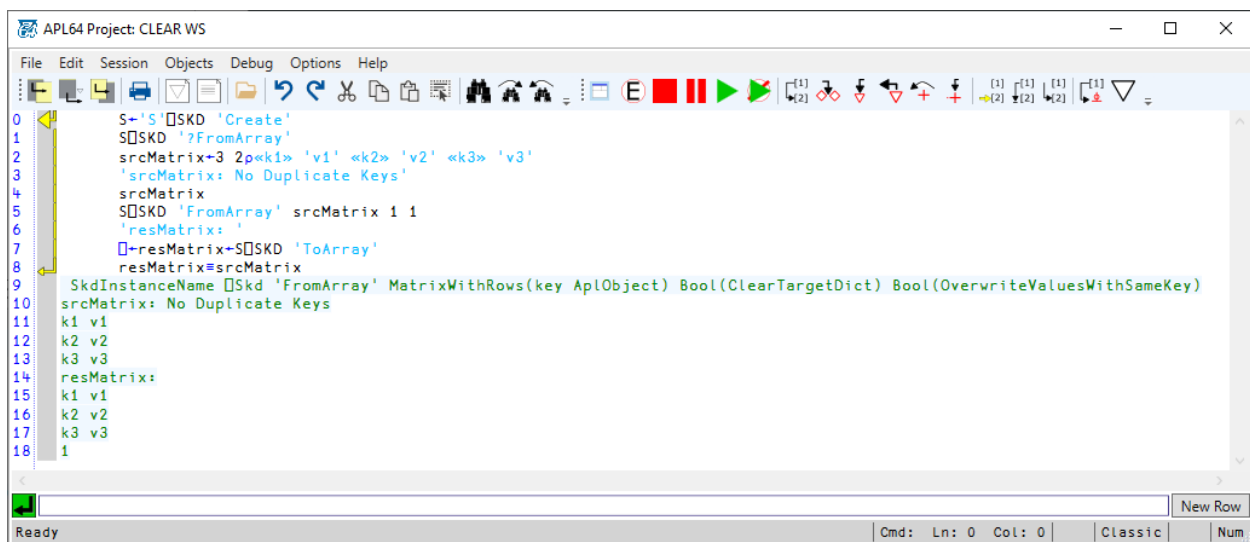


Try It Code:

```
⊞skdself←'S'⊞skd 'Create'
⊞skd 'Add' «abc» 'abcdef'
⊞skd 'ElementAt' 1
⊞skdself≡⊞skd 'self'
```

ToArray

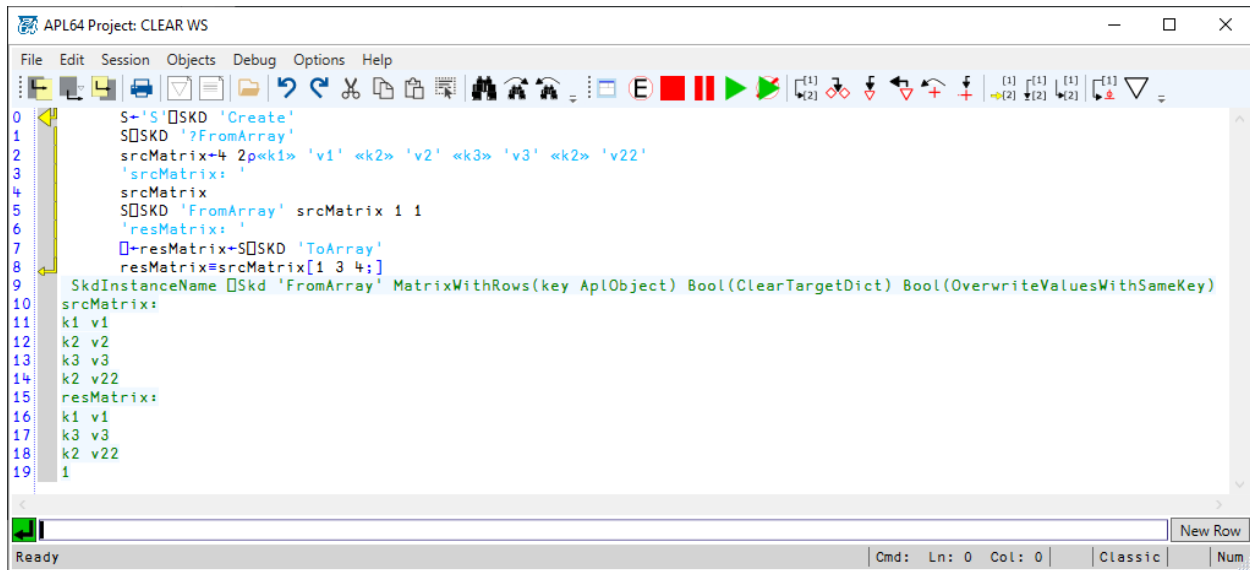
This action is the inverse of the FromArray action. This action returns a matrix with rows the <key, value> pairs of the ⊞skd dictionary instance specified by the left argument.



Try It Code:

```
S←'S'⊞skd 'Create'
S⊞skd '?FromArray'
```

```
srcMatrix←3 2p«k1» 'v1' «k2» 'v2' «k3» 'v3'
'srcMatrix: No Duplicate Keys'
srcMatrix
S⌈skd 'FromArray' srcMatrix 1 1
'resMatrix: '
⌈←resMatrix←S⌈skd 'ToArray'
resMatrix≡srcMatrix
```



Try It Code:

```
S←'S'⌈skd 'Create'
S⌈skd '?FromArray'
srcMatrix←4 2p«k1» 'v1' «k2» 'v2' «k3» 'v3' «k2» 'v22'
'srcMatrix: With duplicate keys'
srcMatrix
S⌈skd 'FromArray' srcMatrix 1 1
'resMatrix: '
⌈←resMatrix←S⌈skd 'ToArray'
resMatrix≡srcMatrix[1 3 4;]
```

ToFileXml

This action will xml serialize the <key, value> pairs in the ⌈skd dictionary instance specified in the left argument and write this serialized data to the file specified in the right argument.

This method is the inverse of the 'FromFileXml' action.

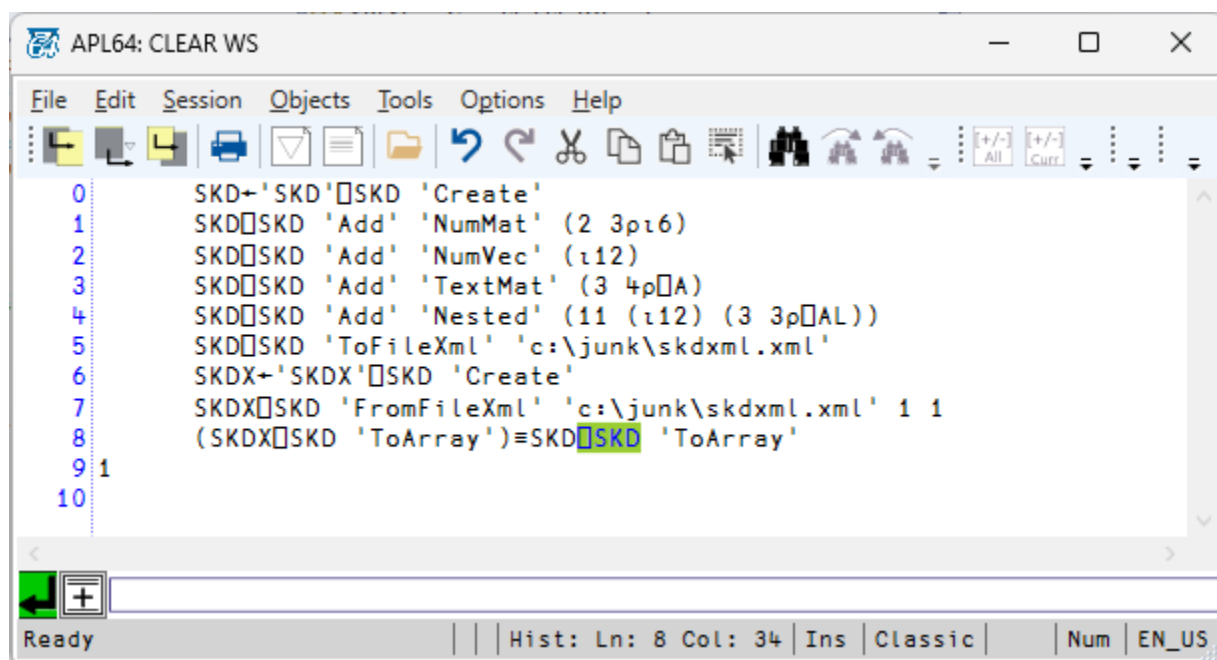
The current user must have read permission to create and write to the specified file.

If the specified file exists, it will be overwritten.

Modification of the serialized data will corrupt the information so that it cannot be deserialized.

Try it Code:

```
SKD←'SKD'⊞SKD 'Create'  
SKD⊞SKD 'Add' 'NumMat' (2 3⍲16)  
SKD⊞SKD 'Add' 'NumVec' (112)  
SKD⊞SKD 'Add' 'TextMat' (3 4⍲A)  
SKD⊞SKD 'Add' 'Nested' (11 (112) (3 3⍲AL))  
SKD⊞SKD 'ToFileXml' 'c:\junk\skdxml.xml'  
SKDX←'SKDX'⊞SKD 'Create'  
SKDX⊞SKD 'FromFileXml' 'c:\junk\skdxml.xml' 1 1  
(SKDX⊞SKD 'ToArray')≡SKD⊞SKD 'ToArray'
```



TryAdd

This action will add the specified <key, value> pair to the specified ☐skd instance dictionary. The right argument of this action is the 'Add' key Aplobject. The key is a character scalar, character vector or string scalar. The Aplobject is the value of the specified <key, value> pair to be added to the ☐skd instance dictionary. The Boolean scalar result indicates if the action was successful.

```

0 S←'S'⊂SKD 'Create'
1 S⊂SKD '?TryAdd'
2 S⊂SKD 'TryAdd' «Item1» (2 3pt6)
3 S⊂SKD 'Count'
4 S⊂SKD 'TryAdd' «Item1» (2 3pt6)
5 BoolScalar ← SkdInstanceName ⊂Skd 'TryAdd' (Key AplObject)
6 1
7 1
8 0

```

Ready | Cmd: Ln: 0 Col: 0 | Classic | Num

Try It Code:

```

S←'S'⊂skd 'Create'
S⊂skd '?TryAdd'
S⊂skd 'TryAdd' «Item1» (2 3pt6)
S⊂skd 'Count'
S⊂skd 'TryAdd' «Item1» (2 3pt6)

```

TryGetValue

This action returns a vector with 1st element a Boolean scalar indicating if the action was successful. If the action is successful the 2nd element of the result is the APL object for the <key, value> pair with key specified in the right argument contained in the `⊂skd` dictionary instance specified in the left argument. If the action fails, e.g. if the key does not exist in the `⊂skd` dictionary instance, the 2nd element of the result is "".

```

0 D1←'D1'⊂SKD 'Create'
1 D1⊂SKD 'Add' «key1» (2 3pt6)
2 D1⊂SKD '?TryGetValue'
3 (1 (2 3pt6))≡D1⊂SKD 'TryGetValue' «key1»
4 (0 '')≡D1⊂SKD 'TryGetValue' «key2»
5 Vector(Bool/Success AplValue) ← SkdInstanceName ⊂Skd 'TryGetValue' KeyText
6 1
7 1

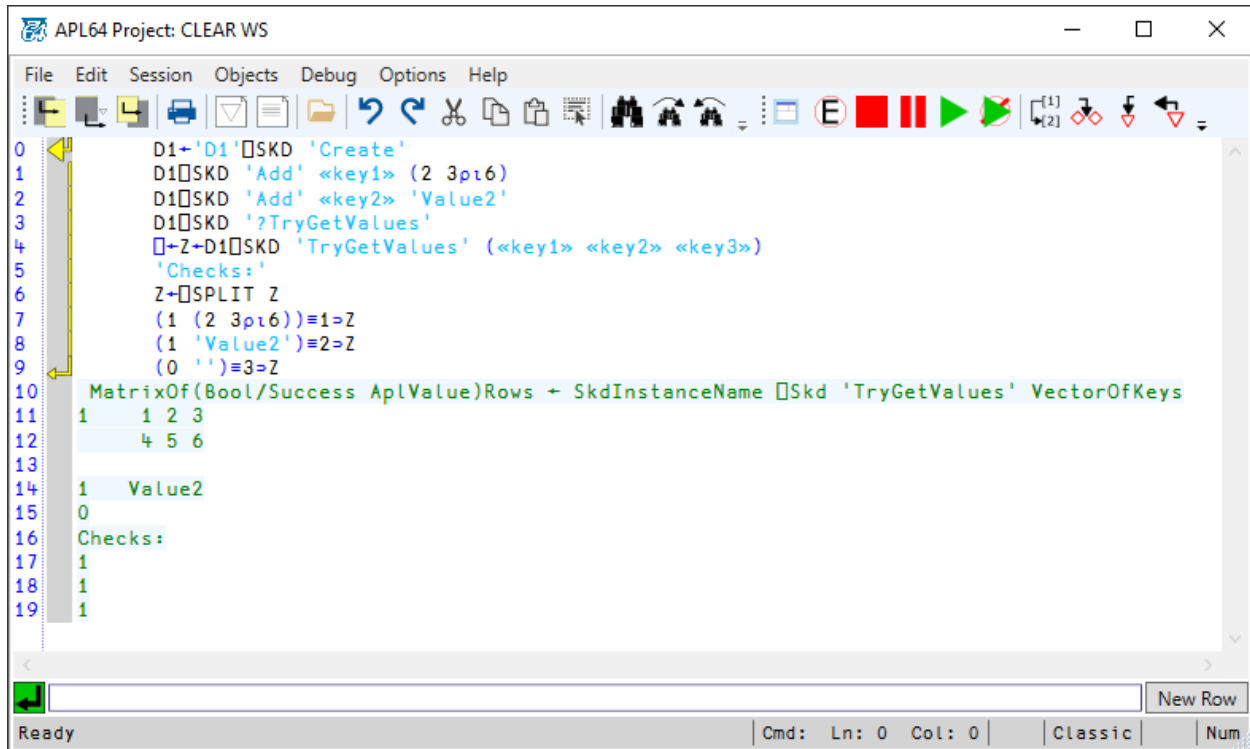
```

Ready | APL Session Command Line: | Num

Try It Code:

```
D1←'D1'□skd 'Create'  
D1□skd 'Add' «key1» (2 3π6)  
D1□skd '?TryGetValue'  
(1 (2 3π6))≡D1□skd 'TryGetValue' «key1»  
(0 '')≡D1□skd 'TryGetValue' «key2»
```

TryGetValues



The screenshot shows the APL64 Project: CLEAR WS interface. The code editor contains the following code:

```
0 D1←'D1'□SKD 'Create'  
1 D1□SKD 'Add' «key1» (2 3π6)  
2 D1□SKD 'Add' «key2» 'Value2'  
3 D1□SKD '?TryGetValues'  
4 □←Z←D1□SKD 'TryGetValues' («key1» «key2» «key3»)  
5 'Checks:'  
6 Z←□SPLIT Z  
7 (1 (2 3π6))≡1⇒Z  
8 (1 'Value2')≡2⇒Z  
9 (0 '')≡3⇒Z  
10 MatrixOf(Bool/Success AplValue)Rows ← SkdInstanceName □Skd 'TryGetValues' VectorOfKeys  
11 1 1 2 3  
12 4 5 6  
13  
14 1 Value2  
15 0  
16 Checks:  
17 1  
18 1  
19 1
```

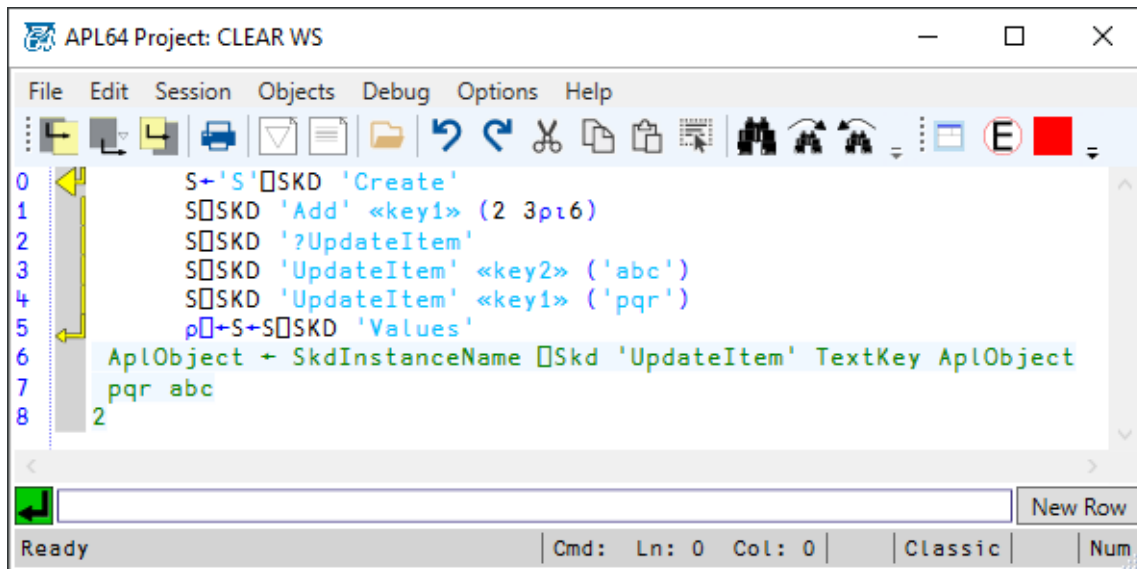
The output window shows the results of the function call, including a matrix of results and a list of checks.

Try It Code:

```
D1←'D1'□skd 'Create'  
D1□skd 'Add' «key1» (2 3π6)  
D1□skd 'Add' «key2» 'Value2'  
D1□skd '?TryGetValues'  
□←Z←D1□skd 'TryGetValues' («key1» «key2» «key3»)  
'Checks:'  
Z←□SPLIT Z  
(1 (2 3π6))≡1⇒Z  
(1 'Value2')≡2⇒Z  
(0 '')≡3⇒Z
```

UpdateItem

This action will replace or add the <key, value> pair in the right argument to the □skd dictionary instance specified in the left argument. The action has no result.



Try It Code:

```

S←'S'⊂skd 'Create'
S⊂skd 'Add' «key1» (2 3⍥6)
S⊂skd '?UpdateItem'
S⊂skd 'UpdateItem' «key2» ('abc')
S⊂skd 'UpdateItem' «key1» ('pqr')
ρ←S←S⊂skd 'Values'

```

Values

This action returns a vector of APL objects which are the values of the <key, value> pairs in the `⎕skd` dictionary instance specified in the left argument.

```
0 S←'S'⎕SKD 'Create'
1 S⎕SKD 'Add' «a» 'abc'
2 S⎕SKD 'Add' «b» ('xyz' (⌈10))
3 S⎕SKD 'Add' «c» 'b'
4 S⎕SKD 'Add' «abc» (2 3⍓6)
5 S⎕SKD '?Keys'
6 ρ←K←S⎕SKD 'Keys'
7 K≡«a» «b» «c» «abc»
8 ρ←V←S⎕SKD 'Values'
9 V≡'abc' ('xyz' (⌈10)) 'b' (2 3⍓6)
10 VectorOfKeys ← SkdInstanceName ⎕Skd 'Keys'
11 a b c abc
12 4
13 1
14 abc xyz 1 2 3 4 5 6 7 8 9 10 b 1 2 3
15 4 5 6
16 4
17 1
```

Try It Code:

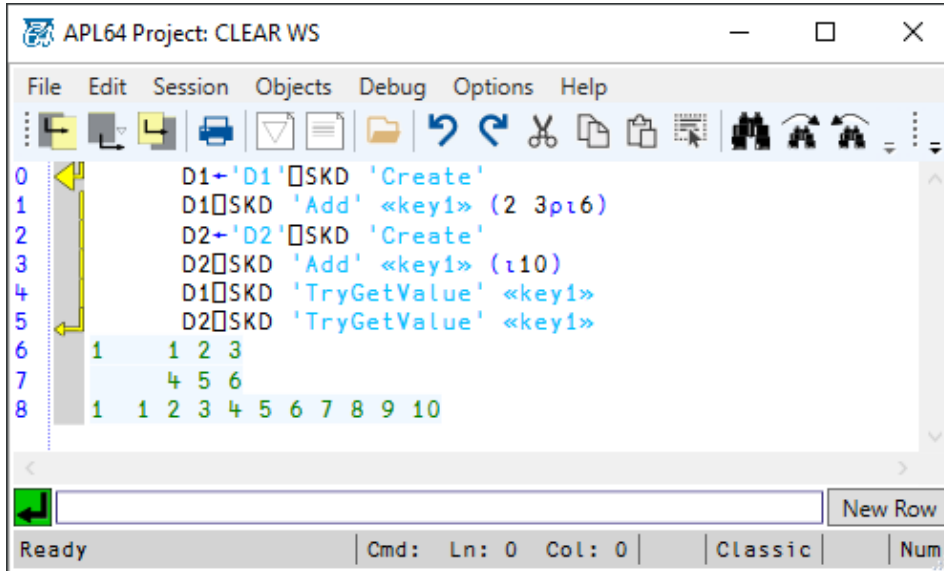
```
S←'S'⎕skd 'Create'
S⎕skd 'Add' «a» 'abc'
S⎕skd 'Add' «b» ('xyz' (⌈10))
S⎕skd 'Add' «c» 'b'
S⎕skd 'Add' «abc» (2 3⍓6)
S⎕skd '?Keys'
ρ←K←S⎕skd 'Keys'
K≡«a» «b» «c» «abc»
ρ←V←S⎕skd 'Values'
V≡'abc' ('xyz' (⌈10)) 'b' (2 3⍓6)
```

Example:

Multiple ☐skd Dictionaries in an APL64 Session

Creating Multiple Dictionaries in the APL64 Session

The ☐skd object can contain multiple, independent ☐skd dictionary instances during an APL64 session.



Try It Code:

```
D1←'D1'⊠skd 'Create'  
D1⊠skd 'Add' «key1» (2 3⍲6)  
D2←'D2'⊠skd 'Create'  
D2⊠skd 'Add' «key1» (110)  
D1⊠skd 'TryGetValue' «key1»  
D2⊠skd 'TryGetValue' «key1»
```