Using ⎕SKD

Contents

□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 is □skd action available by prefixing the Action argument text with '?', e.g.

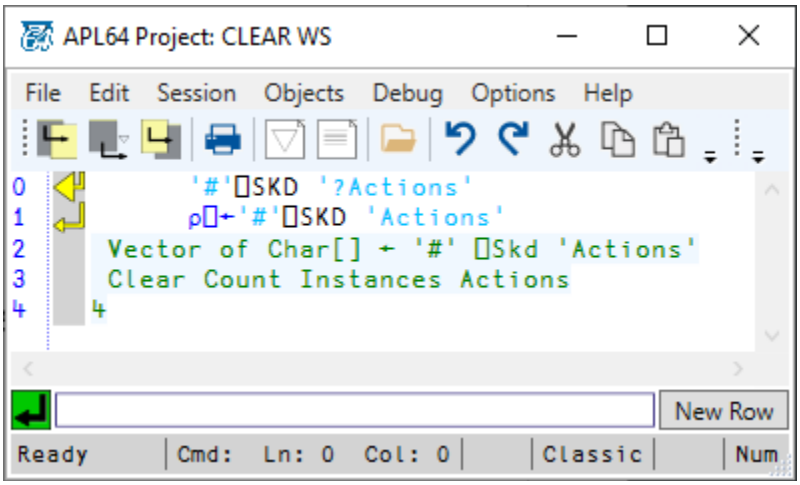
'xyz' □skd '?Clear'

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:

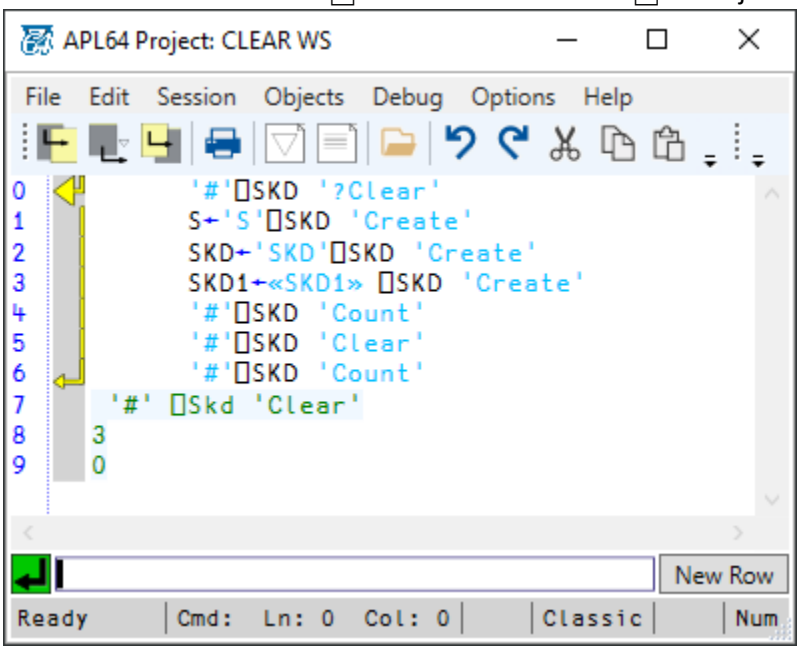


Try It Code:

```
'#'□skd '?Actions'
⍴□←'#'□skd 'Actions'
```

CLEAR

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



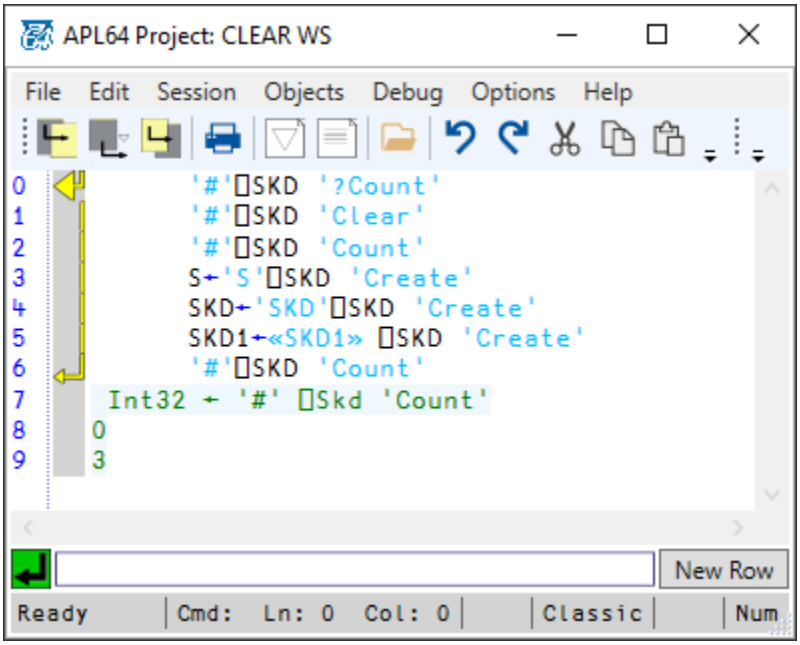
Try It Code:

```
'#'□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.

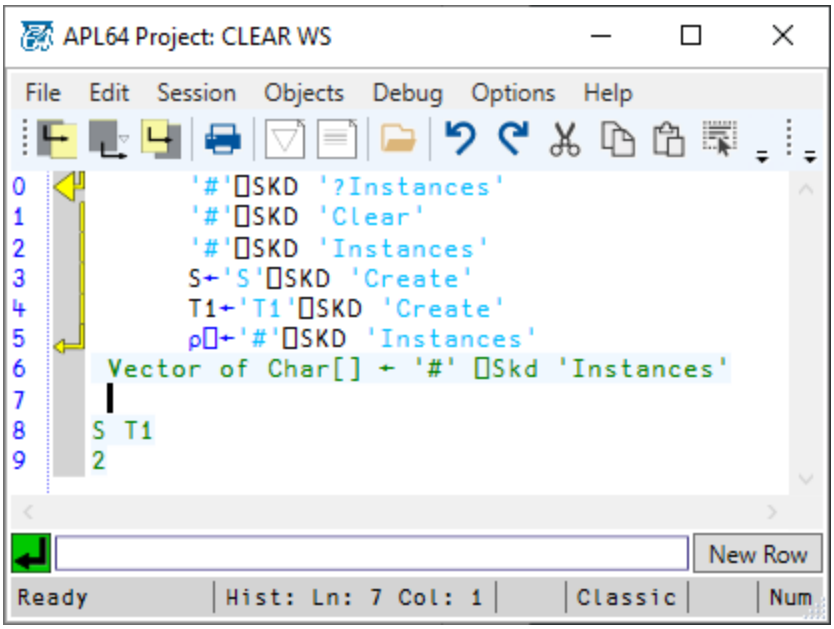


Try It Code:

```
'#'⎕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.

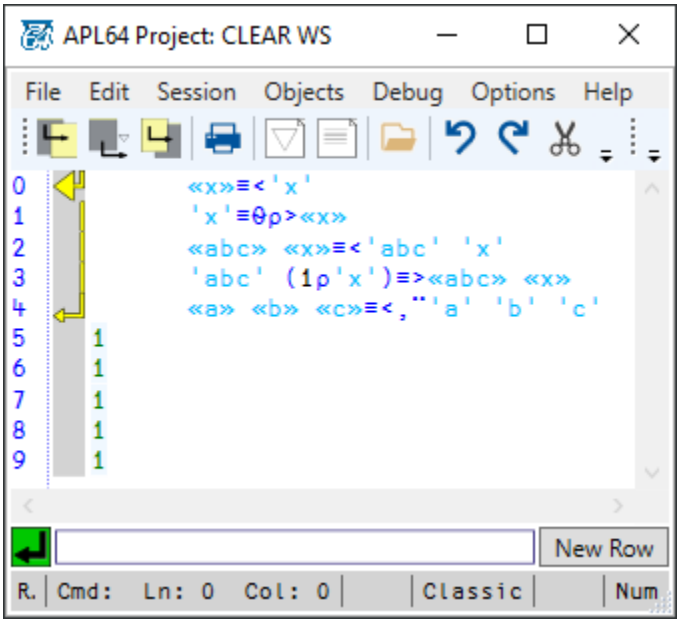


Try It Code:

```
'#'□skd '?Instances'
'#'□skd 'Clear'
'#'□skd 'Instances'
S←'S'□skd 'Create'
T1←'T1'□skd 'Create'
⍴□←'#'□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.



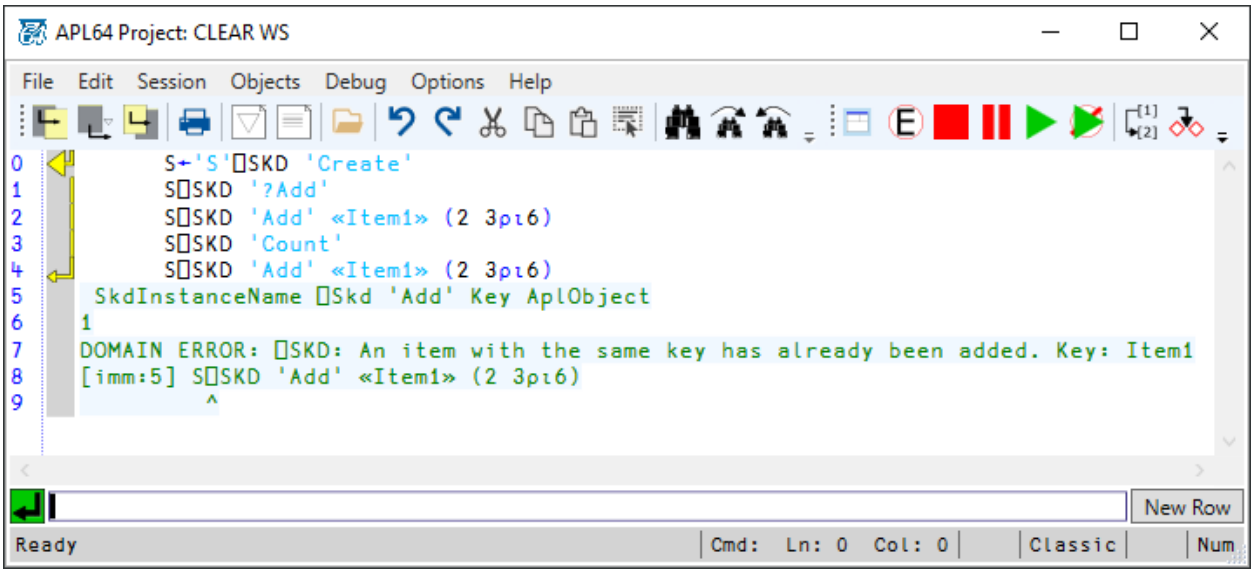
Try Is Code:

```
«x»≡<'x'
'x'≡⍬⍴>«x»
«abc» «x»≡<'abc' 'x'
'abc' (1⍴'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.

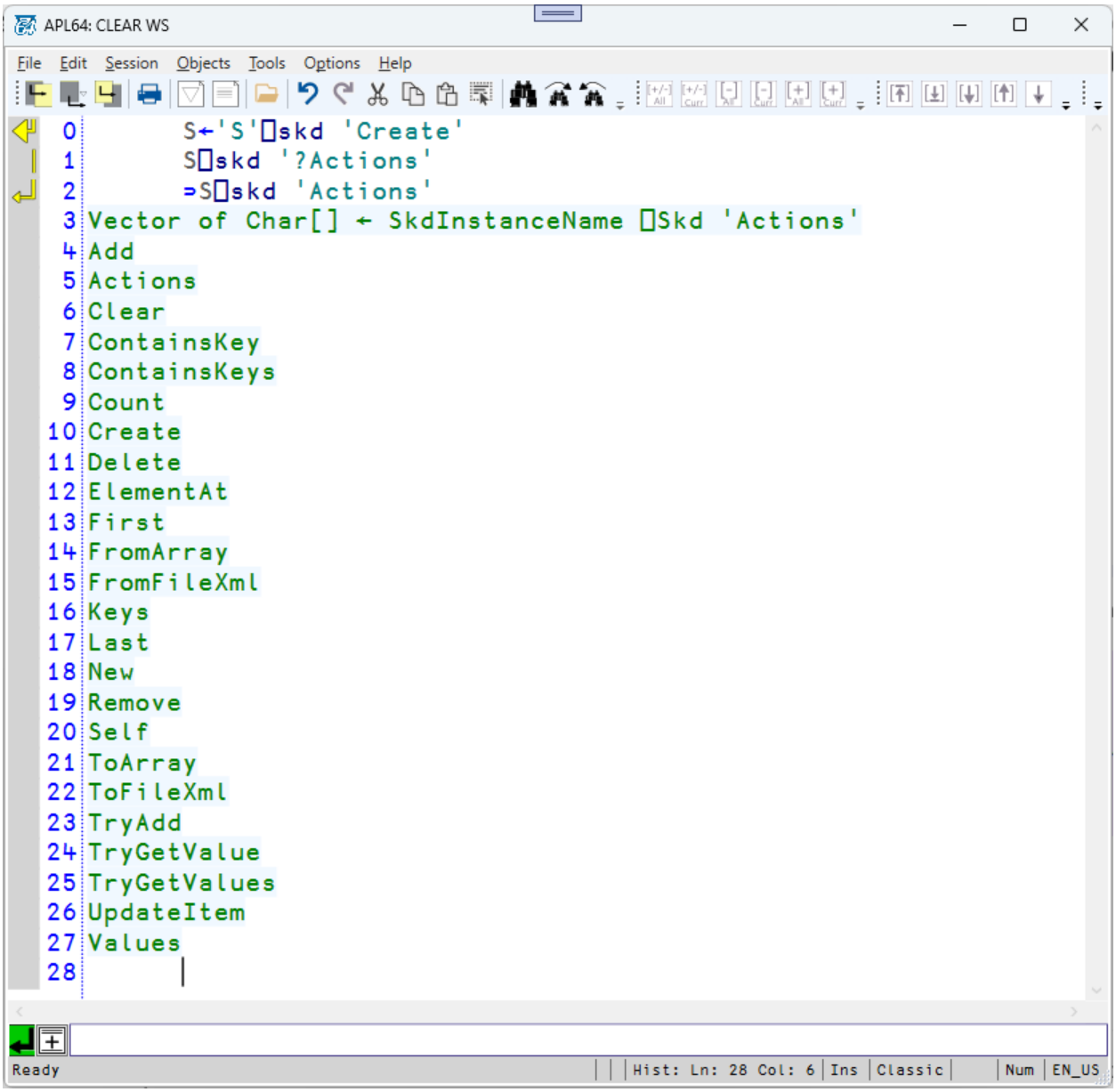


Try It Code:

```
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:

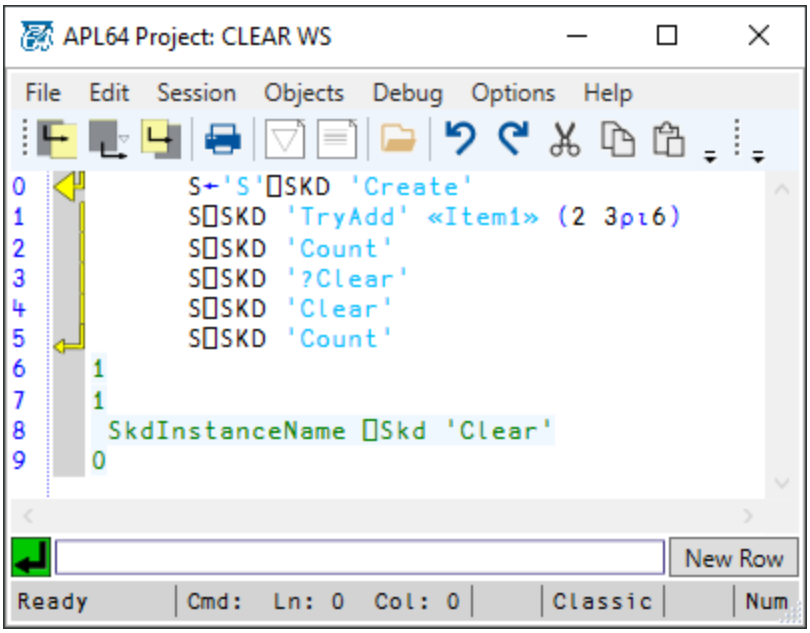


Try It Code:

```
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.

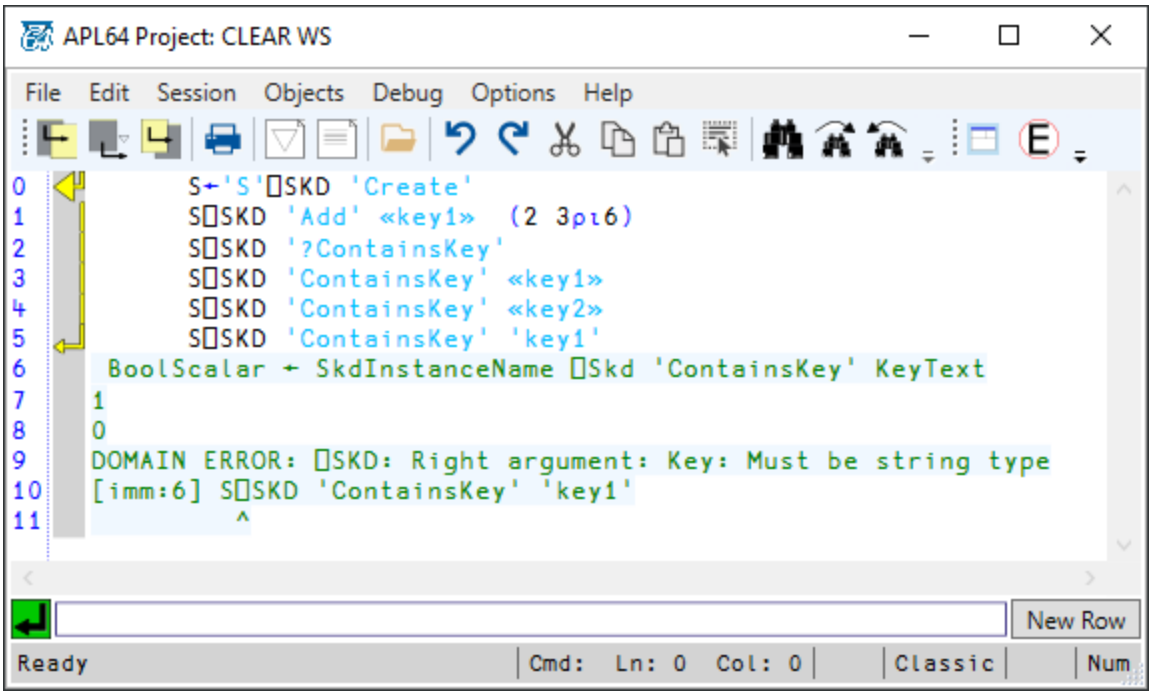


Try It Code:

```
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.

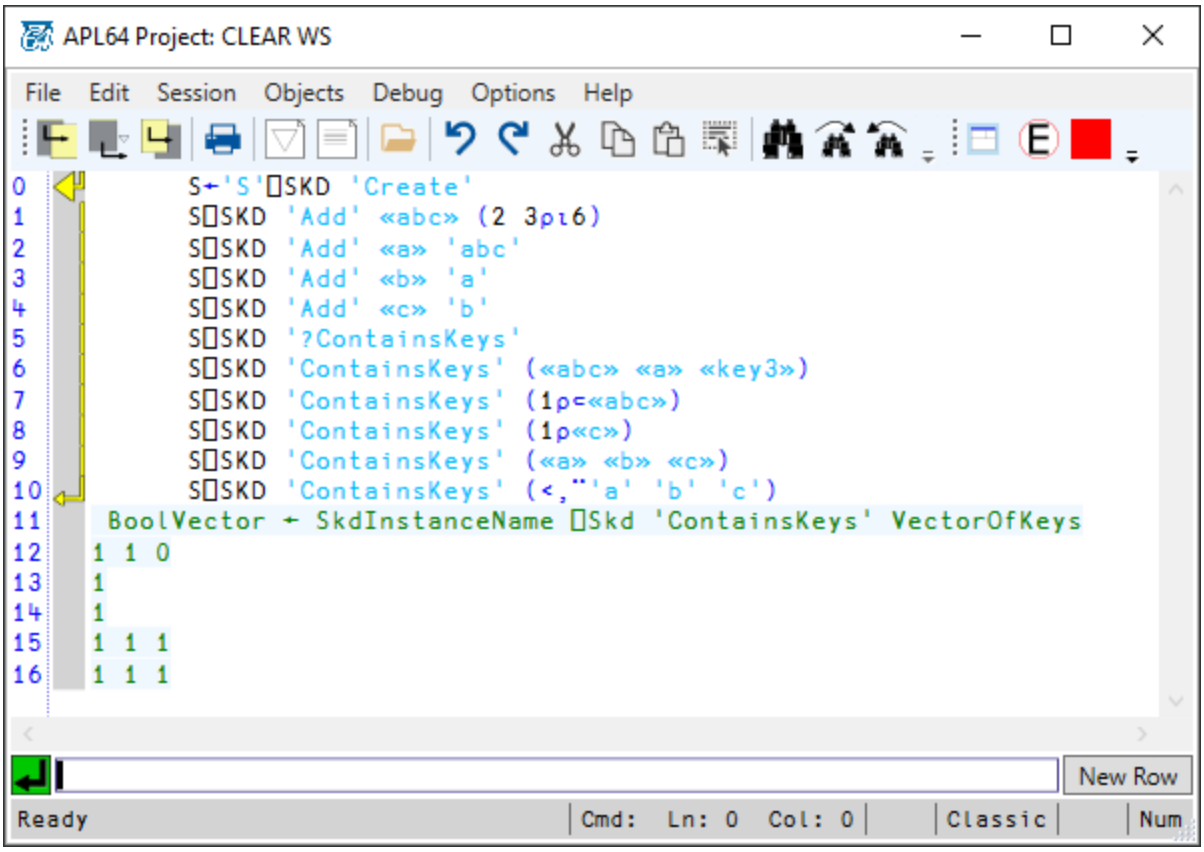


Try It Code:

```
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.

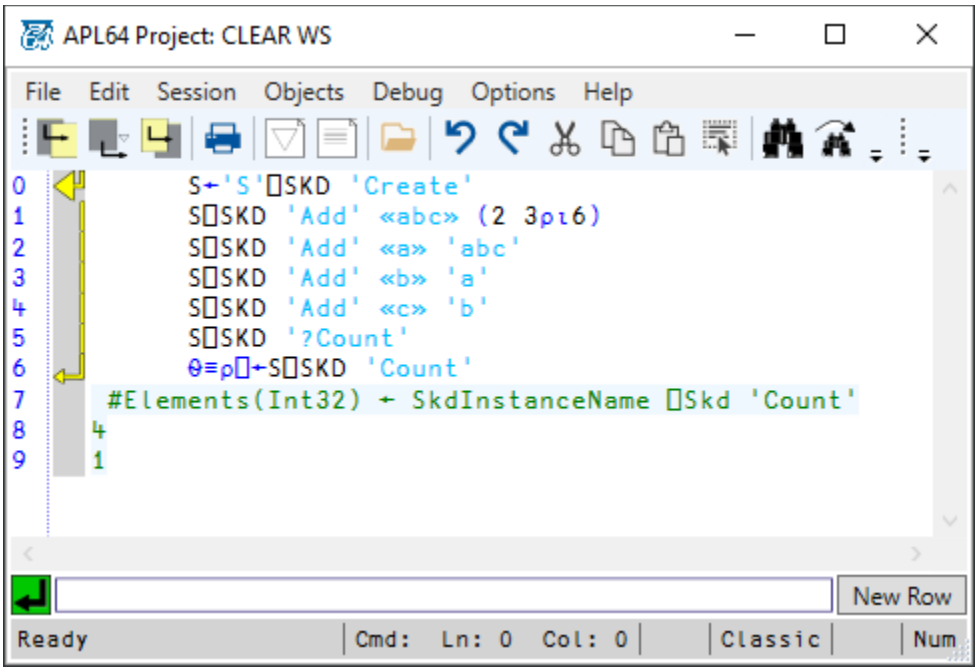


Try It Code:

```
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⍴⍳6)
S□skd 'Add' «a» 'abc'
S□skd 'Add' «b» 'a'
S□skd 'Add' «c» 'b'
S□skd '?Count'
⍬≡⍴□←S□skd 'Count'
```

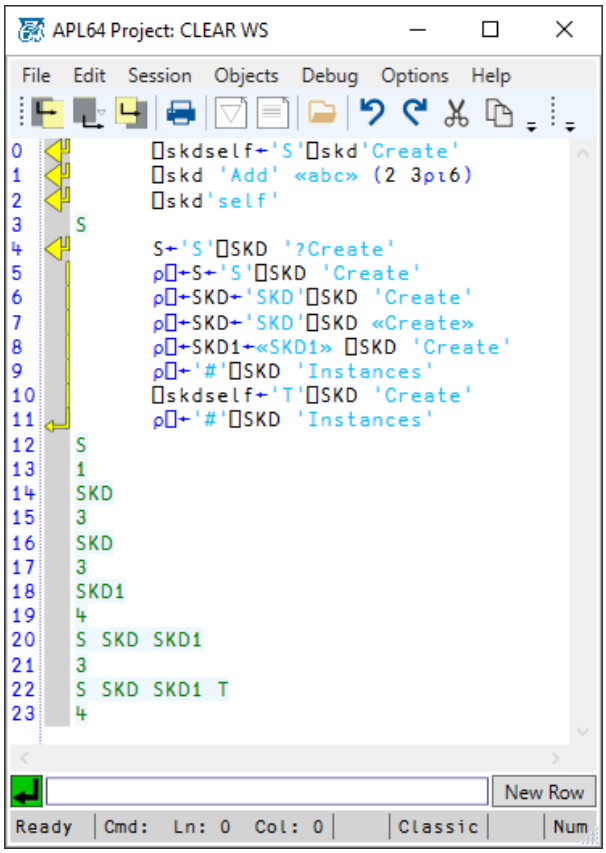
Create

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.

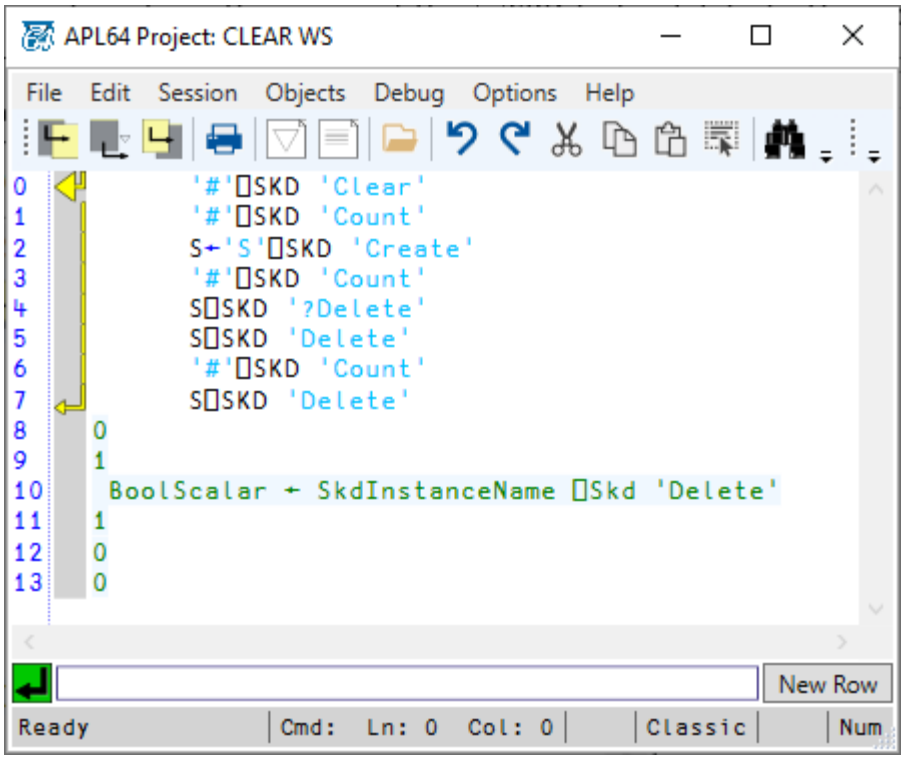


Try It Code:

```
S←'S'□skd '?Create'
⍴□←S←'S'□skd 'Create'
⍴□←skd←'skd'□skd 'Create'
⍴□←skd←'skd'□skd «Create»
⍴□←skd1←«skd1» □skd 'Create'
⍴□←'#'□skd 'Instances'
□skdself←'T'□skd 'Create'
⍴□←'#'□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.

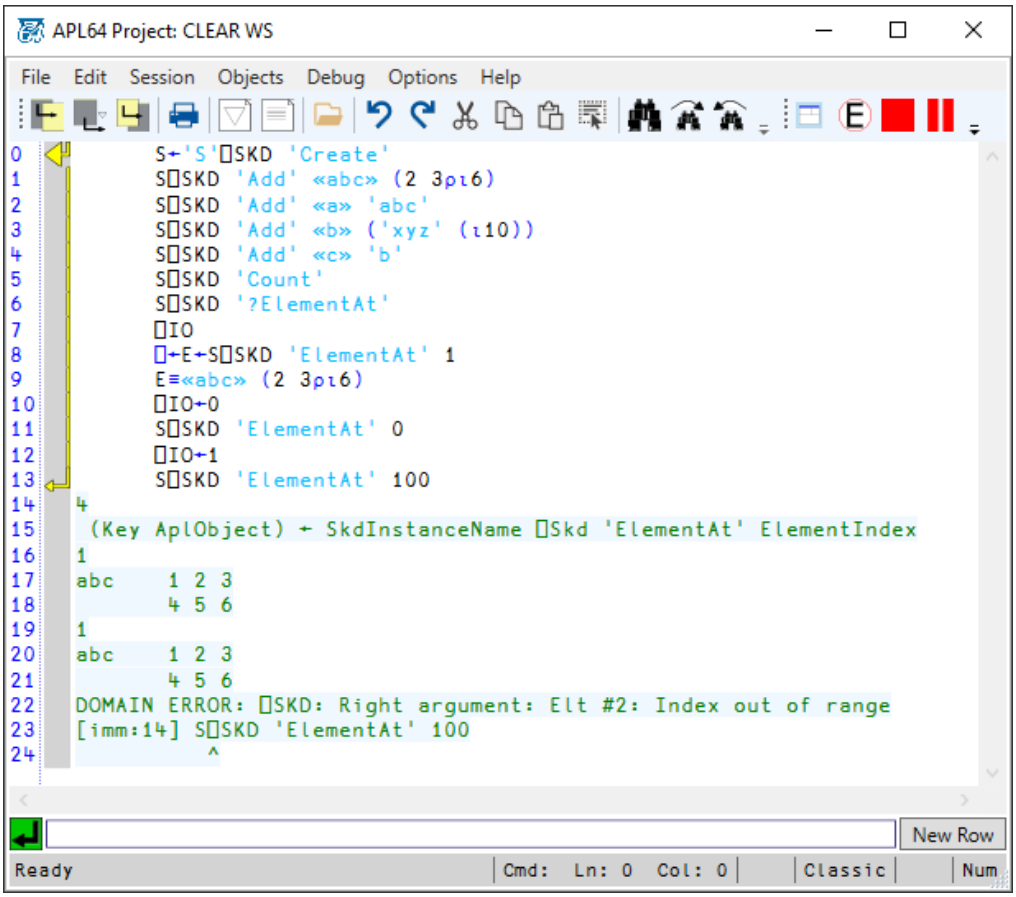


Try It Code:

```
'#'⎕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.

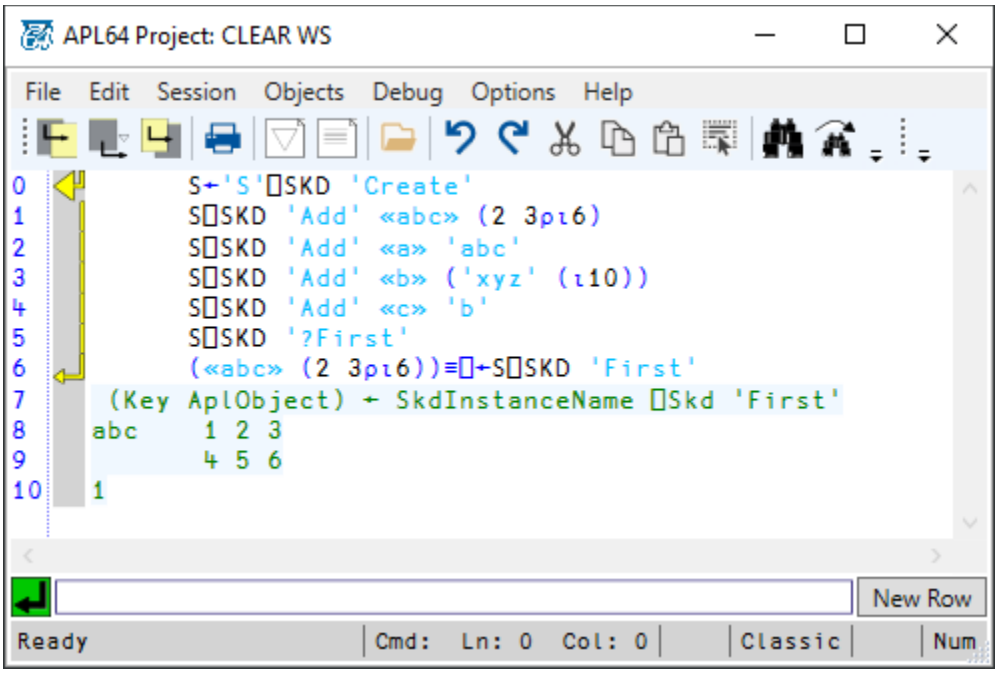


Try It Code:

```
S←'S'⎕skd 'Create'
S⎕skd 'Add' «abc» (2 3⍴⍳6)
S⎕skd 'Add' «a» 'abc'
S⎕skd 'Add' «b» ('xyz' (⍳10))
S⎕skd 'Add' «c» 'b'
S⎕skd 'Count'
S⎕skd '?ElementAt'
⎕io
⎕←E←S⎕skd 'ElementAt' 1
E≡«abc» (2 3⍴⍳6)
⎕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⍴⍳6)
S⎕skd 'Add' «a» 'abc'
S⎕skd 'Add' «b» ('xyz' (⍳10))
S⎕skd 'Add' «c» 'b'
S⎕skd '?First'
(«abc» (2 3⍴⍳6))≡⎕←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.

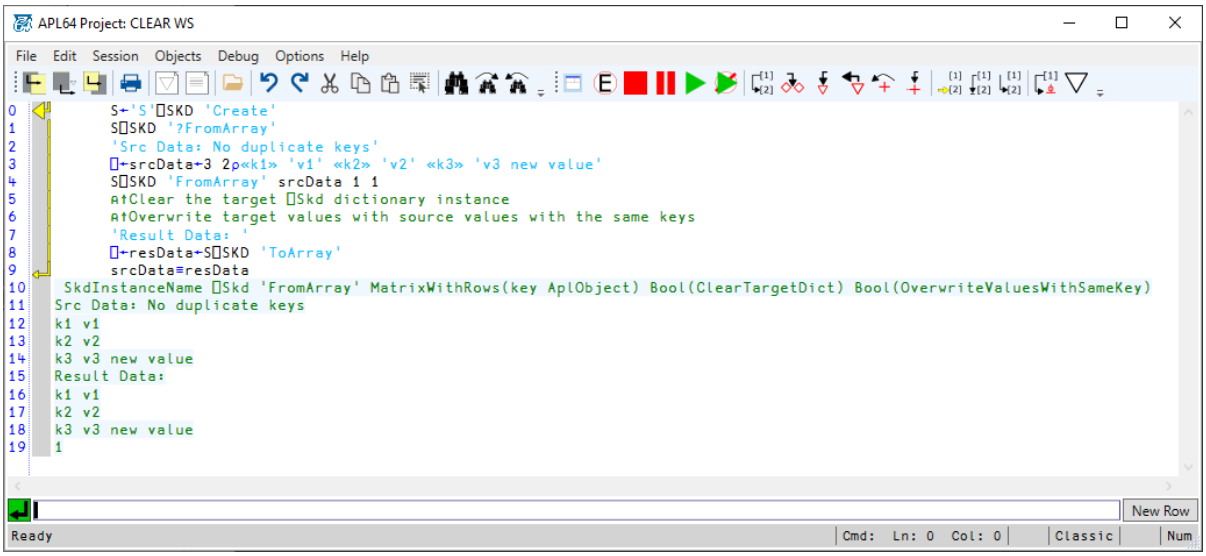
The 2nd right argument (Boolean) indicates if the target ⎕skd dictionary instance should be cleared of any existing <key, value> pairs before adding the source data.

The 3nd right argument (Boolean) indicates if <key, value> pairs in the target ⎕skd dictionary instance should 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.

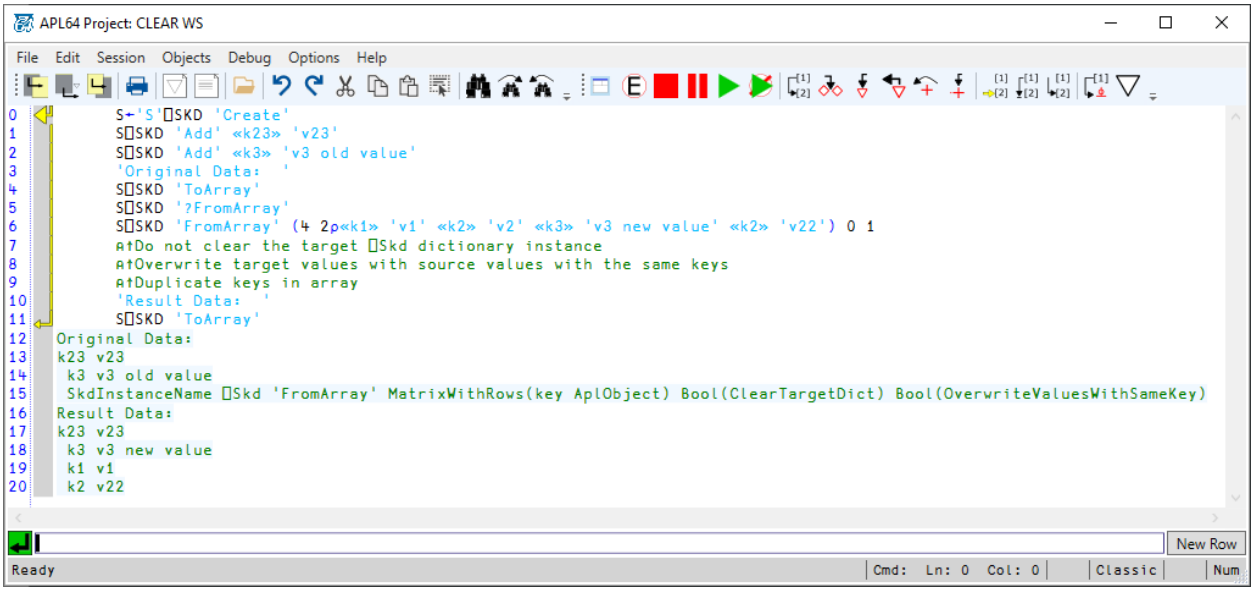
The FromArray action can be used to:

- Replace an □skd dictionary instance
- Merge □skd dictionary instances



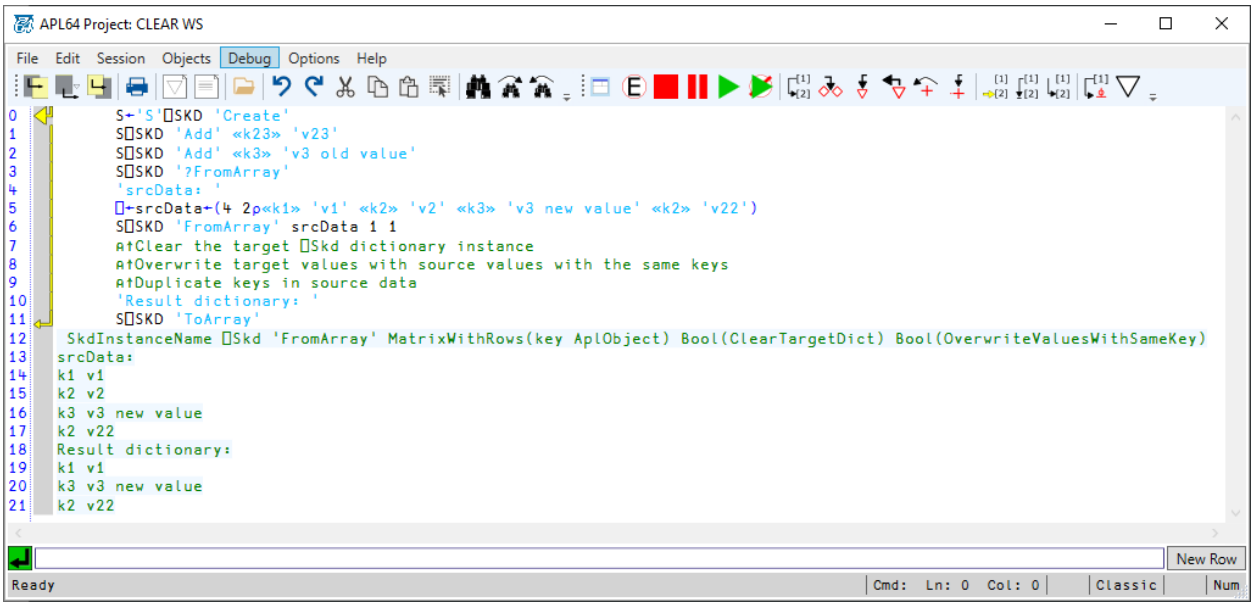
Try It Code:

```
S←'S'□skd 'Create'
S□skd '?FromArray'
'Src Data: No duplicate keys'
□←srcData←3 2⍴«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 2⍴«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 2⍴«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 3nd right argument (Boolean) indicates if <key, value> pairs in the target ⎕skd dictionary instance should overwritten by <key, value> pairs in the source data if the keys are the same.

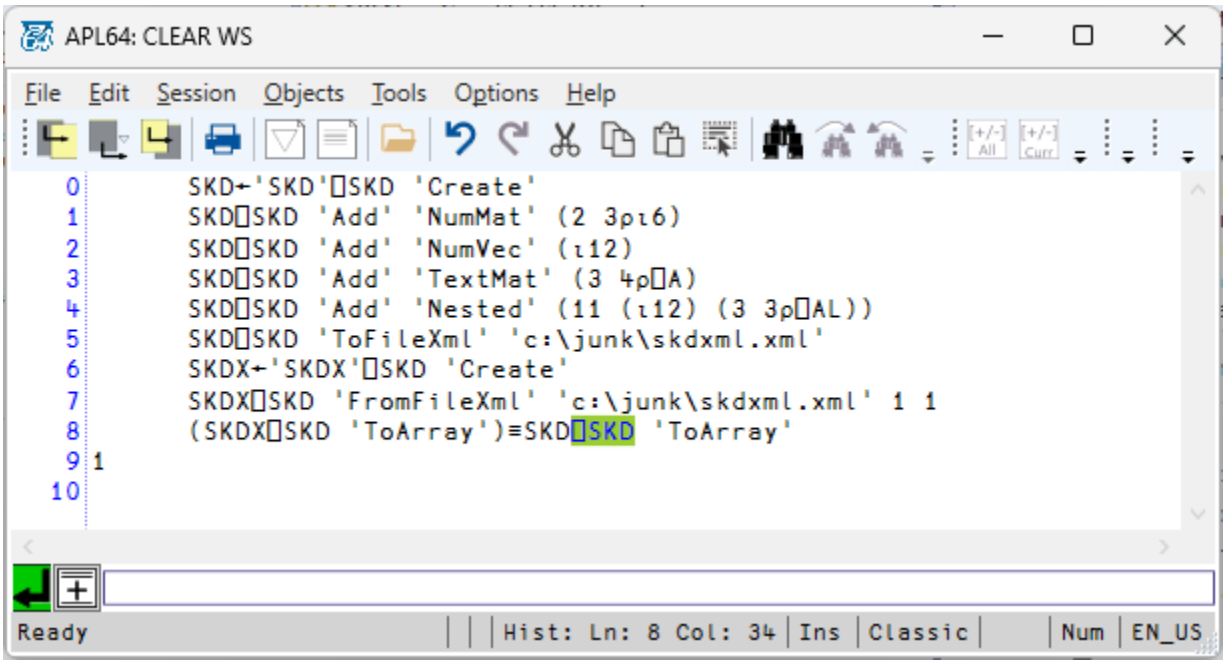
The current user must have read permission for the source data file. Modification of the serialized data will corrupt the information so that it cannot be de-serialized. This action is the inverse of the ToFileXml action.

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:

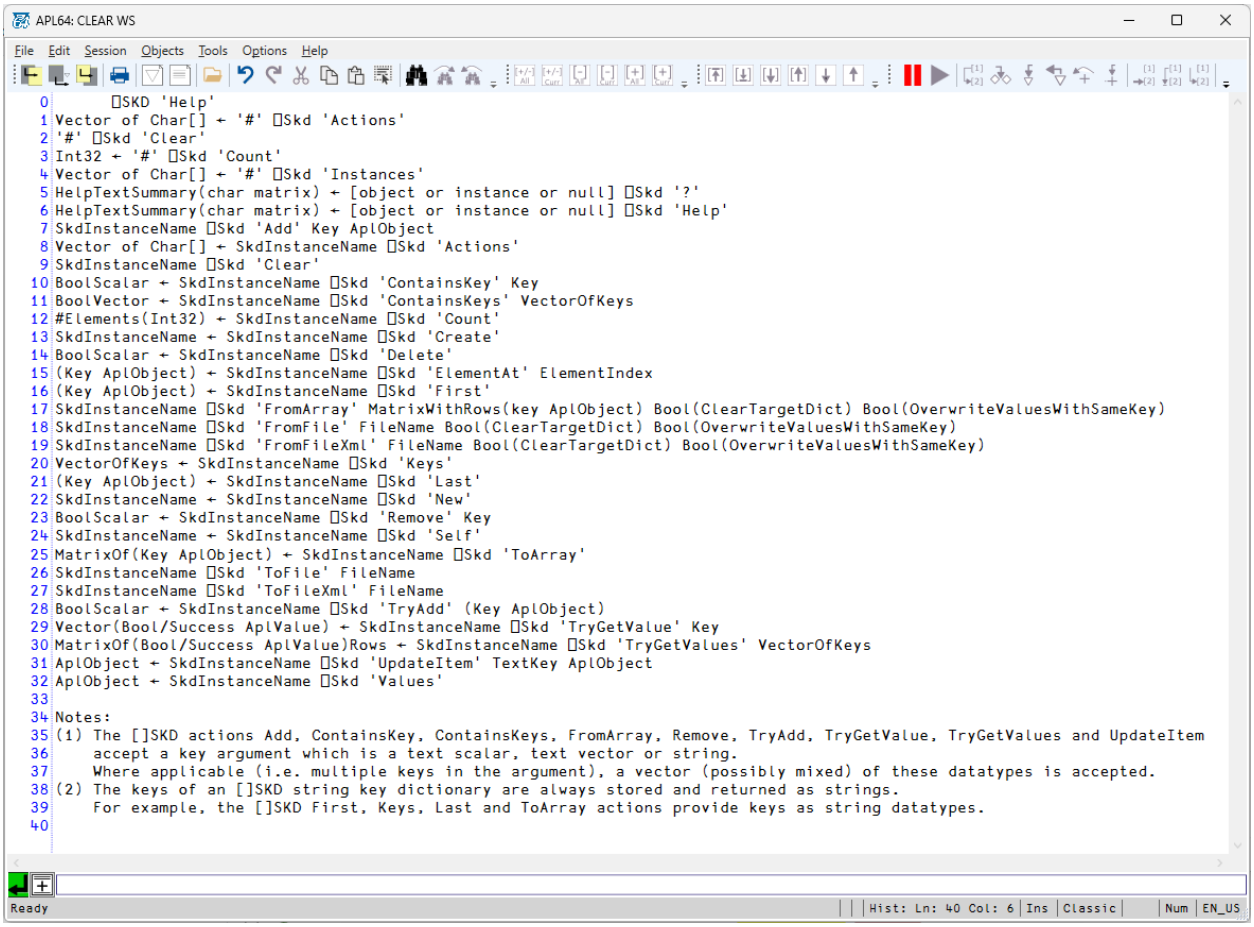
```
SKD←'SKD'⎕SKD 'Create'
SKD⎕SKD 'Add' 'NumMat' (2 3⍴⍳6)
SKD⎕SKD 'Add' 'NumVec' (⍳12)
SKD⎕SKD 'Add' 'TextMat' (3 4⍴⎕A)
SKD⎕SKD 'Add' 'Nested' (11 (⍳12) (3 3⍴⎕AL))
SKD⎕SKD 'ToFileXml' 'c:\junk\skdxml.xml'
SKDX←'SKDX'⎕SKD 'Create'
SKDX⎕SKD 'FromFileXml' 'c:\junk\skdxml.xml' 1 1
(SKDX⎕SKD 'ToArray')≡SKD⎕SKD 'ToArray'
```



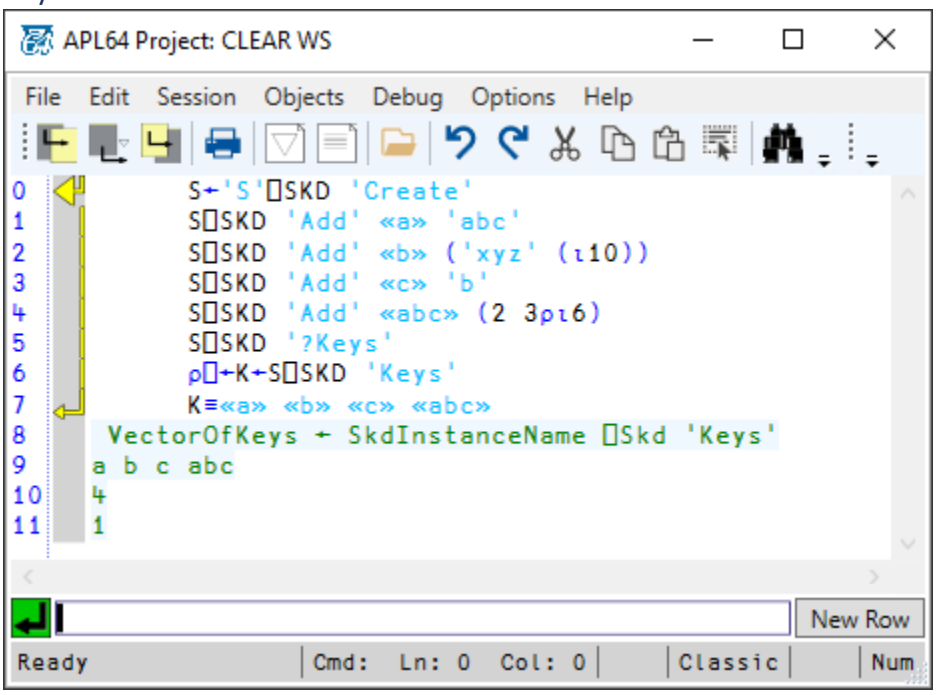
Help

Char[;]←⎕SKD "HELP"

```
⎕skd 'Help'
```



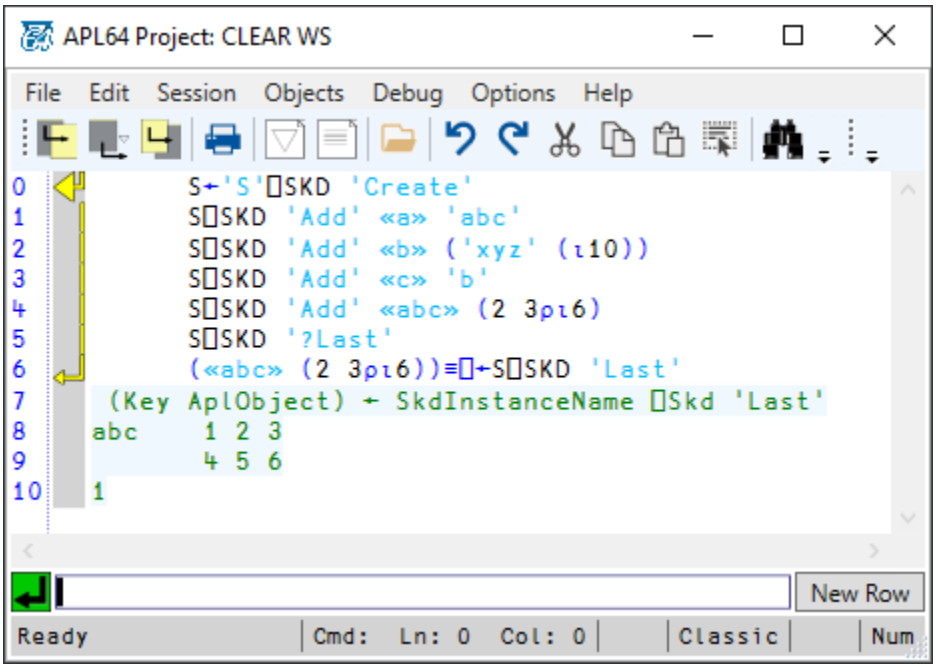
Keys



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'
⍴⎕←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'
```

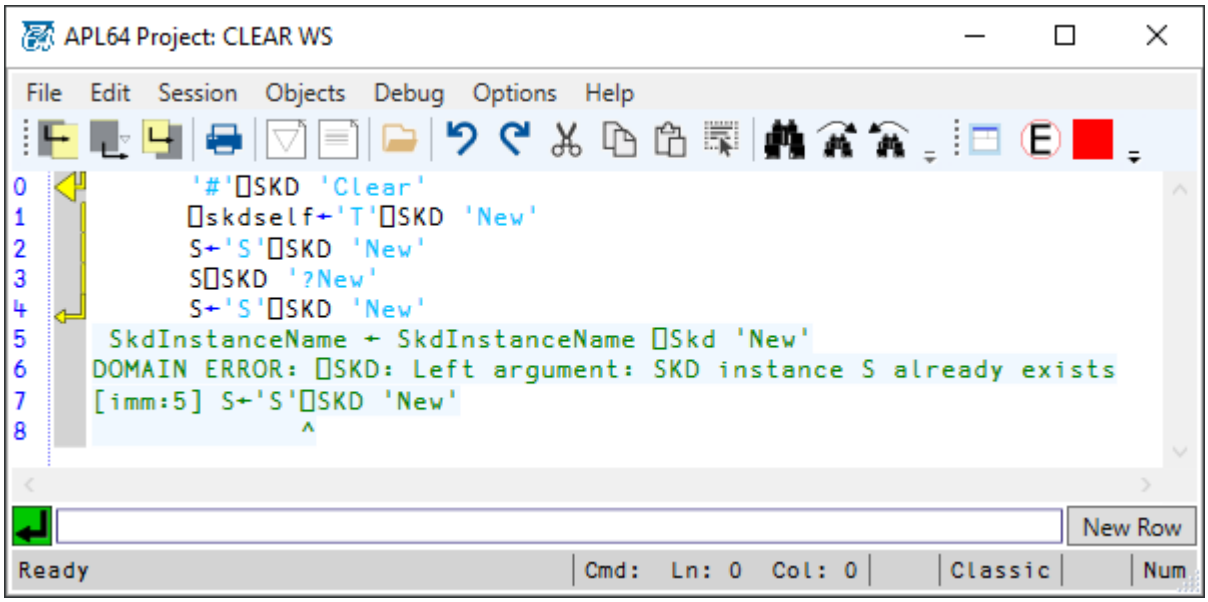
New

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.

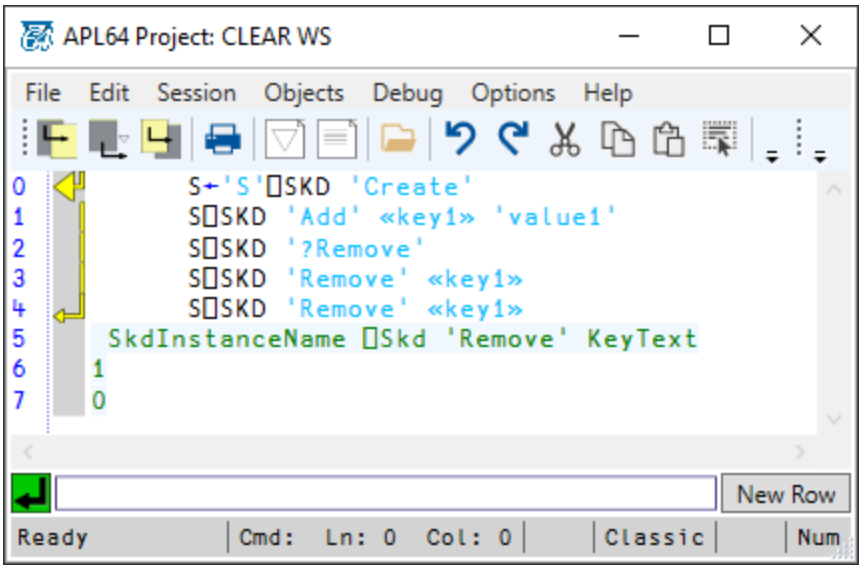


Try It Code:

```
'#'□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 fight argument in the □skd dictionary instance specified in the left argument. The Boolean scalar result indicates if the action was successful.

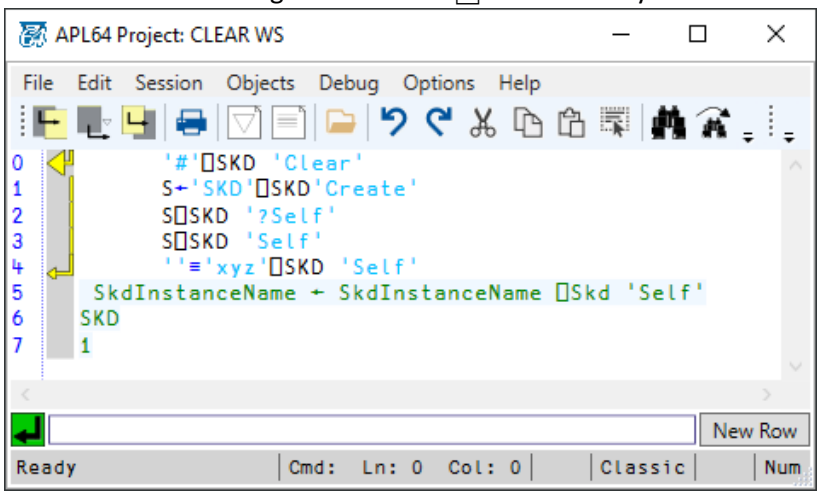


Try It Code:

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

Self

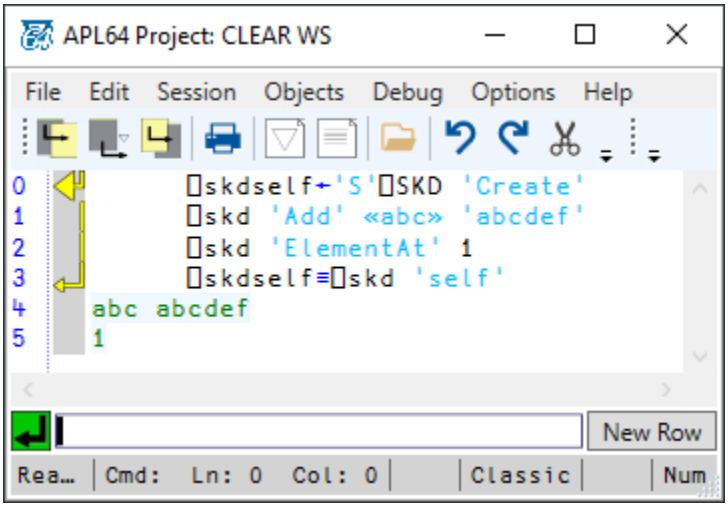
This action returns a character vector containing the name of ⎕skd dictionary instance corresponding to the value of the left argument. If the ⎕skd dictionary instance does not exist, the result ‘’.



Try It Code:

```
'#'⎕skd 'Clear'
S←'skd'⎕skd'Create'
S⎕skd '?Self'
S⎕skd 'Self'
```

```
''≡'xyz'⎕skd 'Self'
```

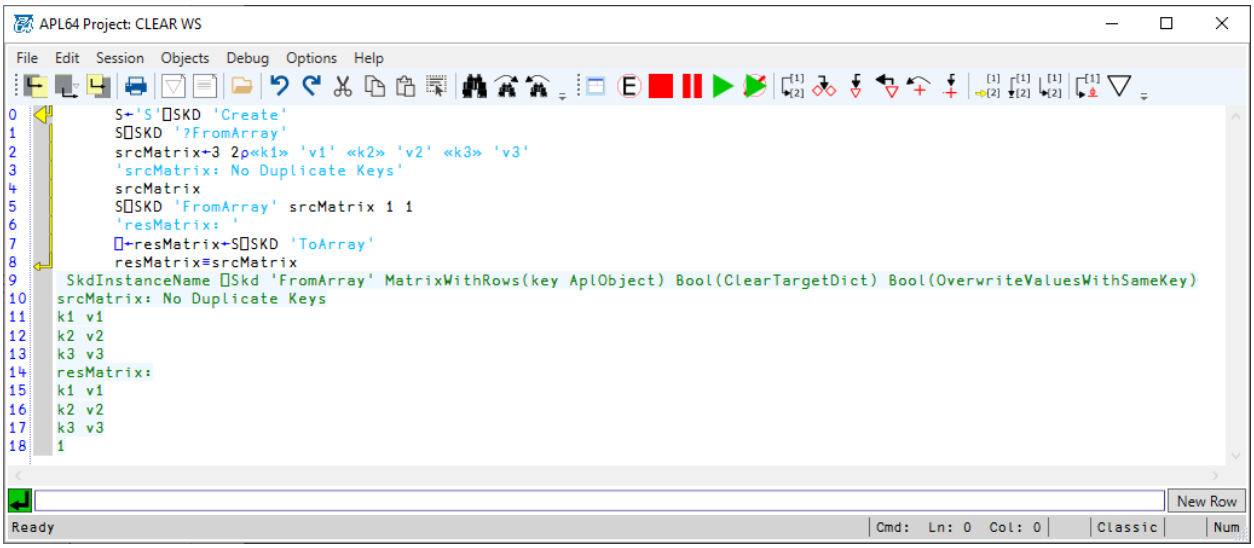


Try It Code:

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

ToArray

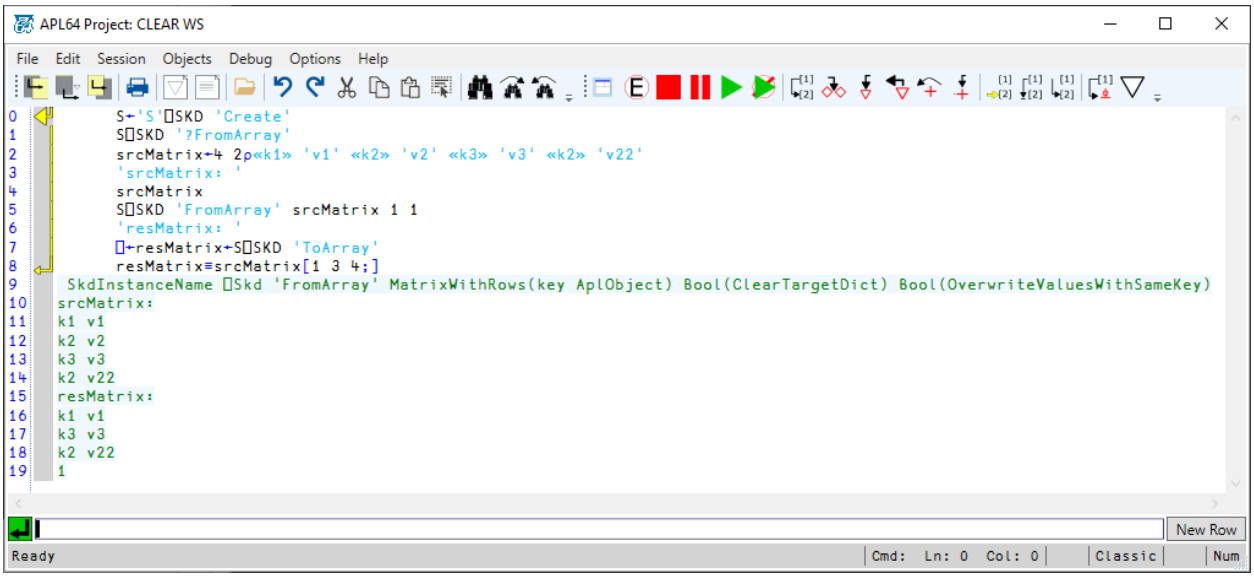
This action is the inverse of the FromArray action. This action returns a matrix with rows the <key, value> pairs of the ⎕skd dictionary instance specified by the left argument.



Try It Code:

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

```
srcMatrix←3 2⍴«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 2⍴«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.

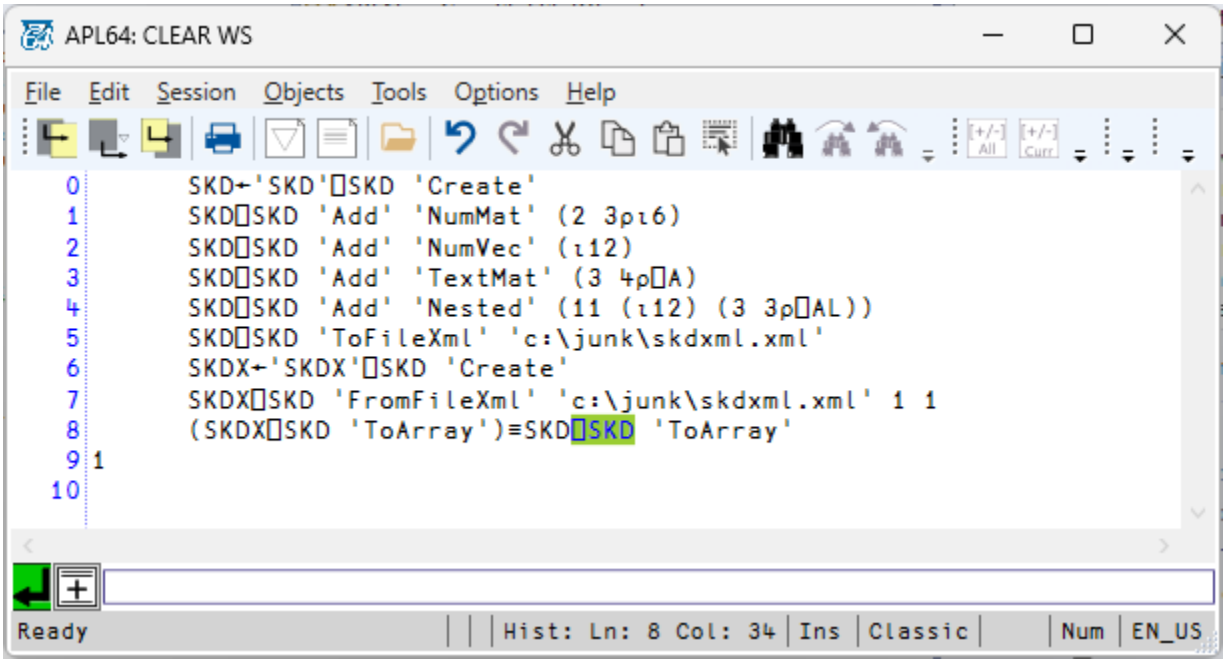
The current user must have read permission to create and write to the specified file.

If the specified file exists, it will be overwritten.

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

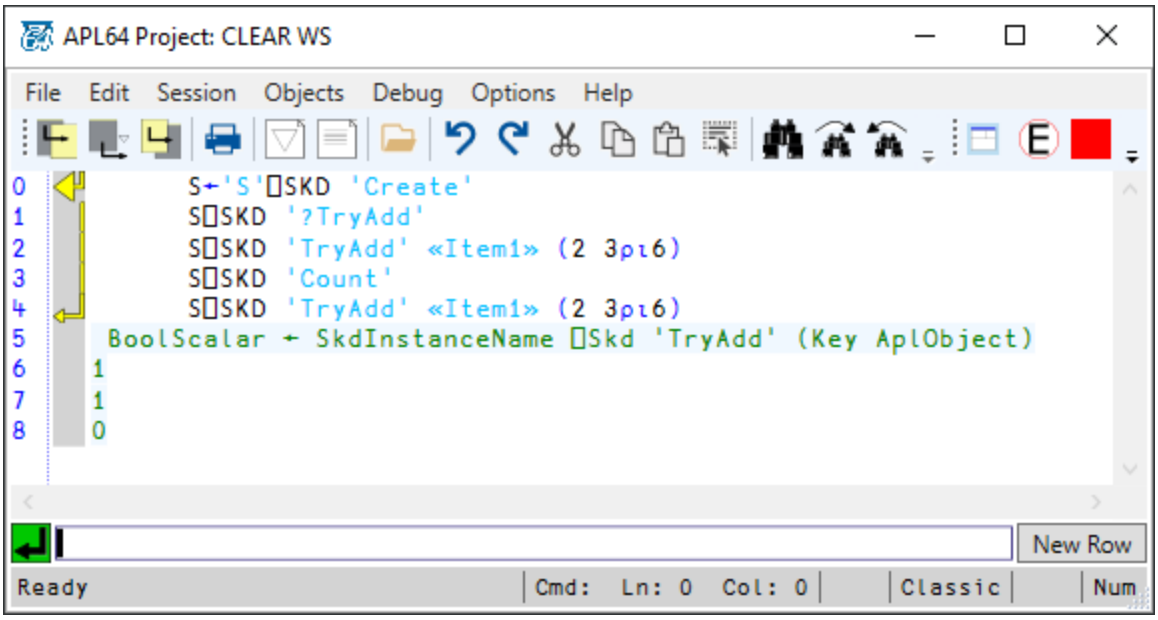
Try it Code:

```
SKD←'SKD'⎕SKD 'Create'
SKD⎕SKD 'Add' 'NumMat' (2 3⍴⍳6)
SKD⎕SKD 'Add' 'NumVec' (⍳12)
SKD⎕SKD 'Add' 'TextMat' (3 4⍴⎕A)
SKD⎕SKD 'Add' 'Nested' (11 (⍳12) (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.

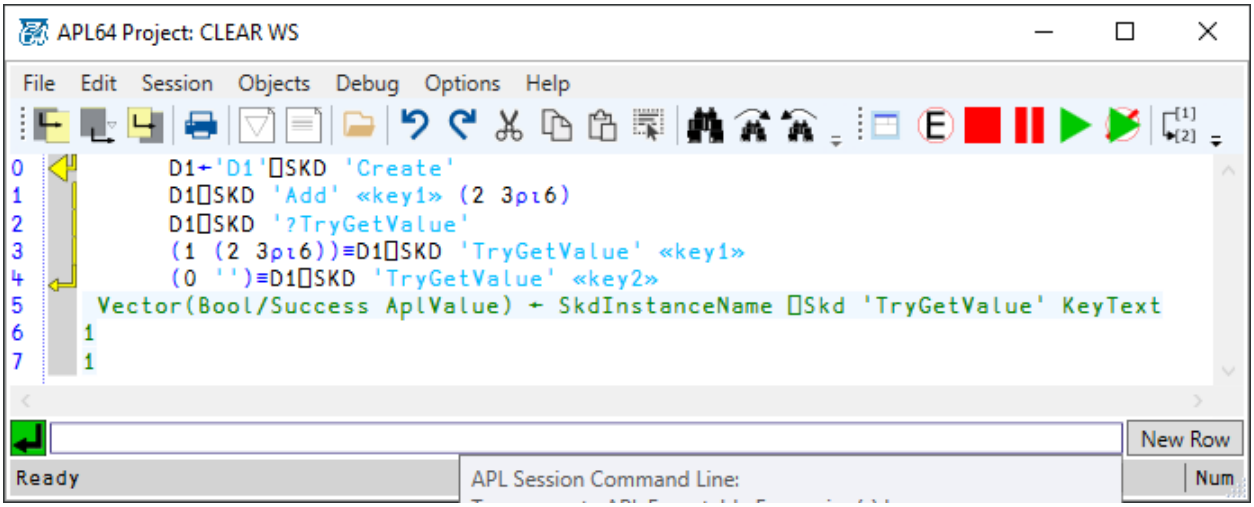


Try It Code:

```
S←'S'⎕skd 'Create'
S⎕skd '?TryAdd'
S⎕skd 'TryAdd' «Item1» (2 3⍴⍳6)
S⎕skd 'Count'
S⎕skd 'TryAdd' «Item1» (2 3⍴⍳6)
```

TryGetValue

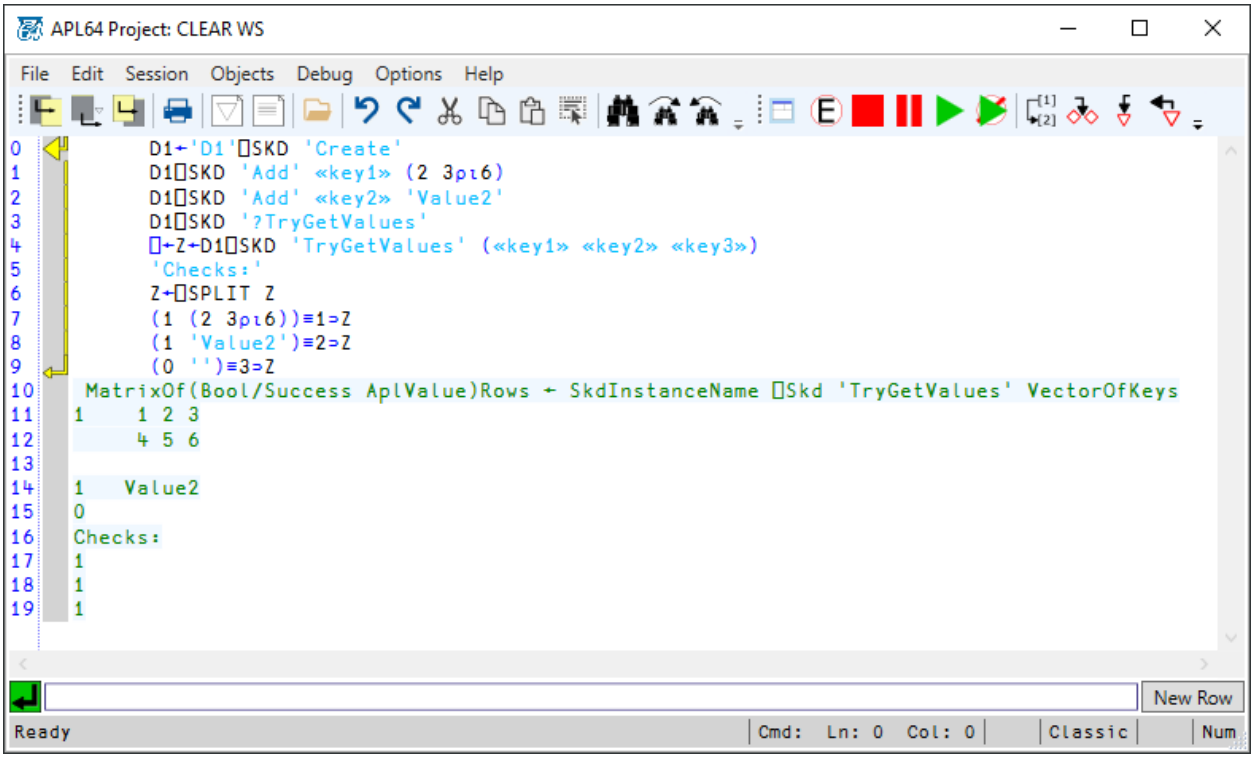
This action returns a vector with 1st element a Boolean scalar indicating if the action was successful. If the action is successful the 2nd element of the result is the APL object for the <key, value> pair with key specified in the right argument contained in the ⎕skd dictionary instance specified in 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 ''.



Try It Code:

```
D1←'D1'⎕skd 'Create'
D1⎕skd 'Add' «key1» (2 3⍴⍳6)
D1⎕skd '?TryGetValue'
(1 (2 3⍴⍳6))≡D1⎕skd 'TryGetValue' «key1»
(0 '')≡D1⎕skd 'TryGetValue' «key2»
```

TryGetValues

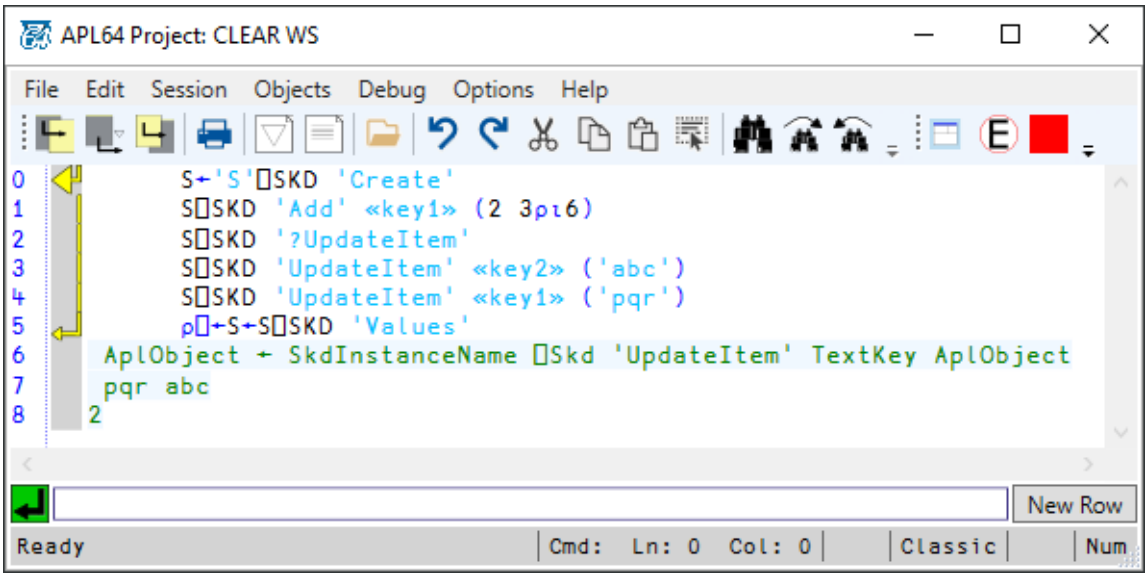


Try It Code:

```
D1←'D1'⎕skd 'Create'
D1⎕skd 'Add' «key1» (2 3⍴⍳6)
D1⎕skd 'Add' «key2» 'Value2'
D1⎕skd '?TryGetValues'
⎕←Z←D1⎕skd 'TryGetValues' («key1» «key2» «key3»)
'Checks:'
Z←⎕SPLIT Z
(1 (2 3⍴⍳6))≡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.

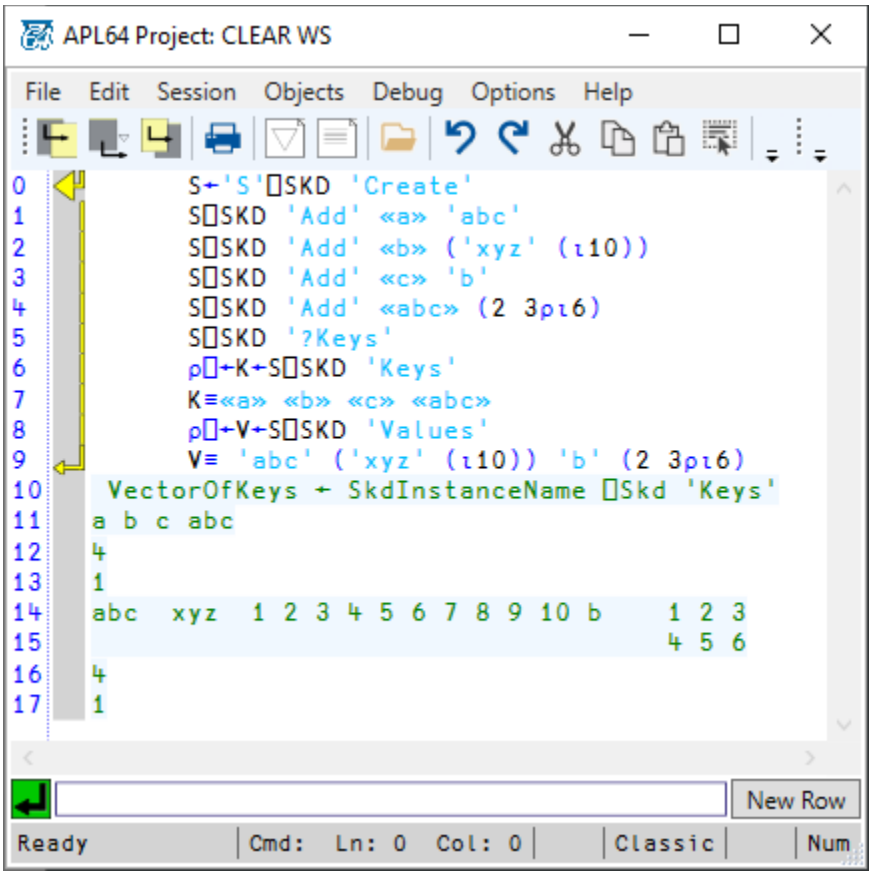


Try It Code:

```
S←'S'⎕skd 'Create'
S⎕skd 'Add' «key1» (2 3⍴⍳6)
S⎕skd '?UpdateItem'
S⎕skd 'UpdateItem' «key2» ('abc')
S⎕skd 'UpdateItem' «key1» ('pqr')
⍴⎕←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.



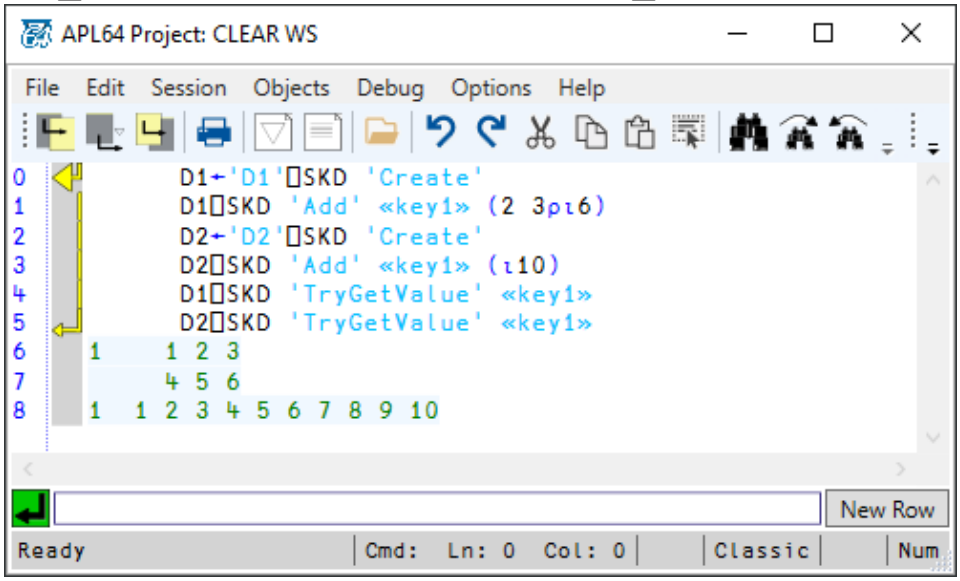
Try It Code:

```
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'
⍴⎕←K←S⎕skd 'Keys'
K≡«a» «b» «c» «abc»
⍴⎕←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» (⍳10)
D1□skd 'TryGetValue' «key1»
D2□skd 'TryGetValue' «key1»
```