

Draft American National Standard for Information  
Systems - Programming Languages - Smalltalk





## **Forward**

Smalltalk is designed to be a "single paradigm language with very simple semantics and syntax for specifying elements of a system and for describing system dynamics." The principle is explained

## **1. Goals and Scope**









defined by the text of a Smalltalk program. Typically these are either *literals* or *class objects*. Some statically created objects are bound to an object name within some scope. Such objects are called *named objects*. The most commonly occurring named objects are class objects.

Dynamically created objects are not individually defined by the program, instead they are dynamically created as a side effect of the execution of a method. Dynamically created objects do not have names. They are typically referenced as the value of a variable.









union of the set of instance variable names of the <<class definition>> and the complete instance variable set of the class definition's superclass. If a <<superclass name>> is not specified in a

















<literal> |





provides the receiver consists solely of the reserved identifier 'self', and the receiver is the class object of a class whose instance objects are indexable or byte indexable the following actions are



### **3.4.6.3 String Literals**

String I







### 3.5.6 Numbers

Numbers are tokens that represent numeric quantities. There are three forms of numbers: integer, float, and scaledDecimal. No white space is allowed within a numeric token.

---

integer ::= decimalInteger | radixInteger

**Rationale**

---













certain operations have been traditionally grouped together. For greater clarity, our goal in defining

### **5.1.2.2 Parameter Specification**

applies to all such values. Multiple return value specifications are required for cases where a message is defined to return objects conforming to different protocols, on a case-specific basis. These are conveniently described with separate conformance statements and aliasing annotations. In order to establish correspondence between sets of return value specifications, we do not permit two distinct return value specifications which promise conformance to the same protocol.

If a message specification has no return value specification (that is, the return value is not











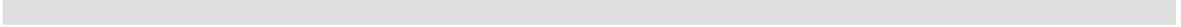
## **5.2 Standard Globals**

The following global values exist with the named protocols in Standard-conforming









receiver == comparand







candidateClass      <Object>      uncaptured

**Return Value**

<boolean>      unspecified

**Errors**

none





**5.3.1.19 Message: respondsTo: selector**

**Synopsis**

### **5.3.2.1 Message Refinement: `printString`**

#### **Synopsis**





**5.3.3.7 Message: ifTrue: operand**

**Synopsis**



none

#### **5.3.4 Protocol: <Character>**

##### **Conforms To**

<Object>

##### **Description**

This protocol describes the behavior that is common to character objects. Character objects serve as the element value fo-13.2IT St-1301(u)-24.8(ellta)la enga.Tre ellta-13.7(o)luag-12.35(epr)-6.2(o)-02.35(v3.46i)-9.2

`receiver = comparand`

is *true* then the receiver and comparand must have *equivalent hash values*. Or more fro-12a7(y)1712.:/F1 1 Tf10 0



Test whether the receiver is a letter or digit.

**Definition:** <Character>







**Definition: <failedMessage>**

Return a collection containing the arguments of the message that could not be sent. The elements of the collection are ordered, from the first element to the last element, in the same order as the arguments of the message, from left to right. If the message had no arguments, the collection will be empty.

**Return Value**

<sequenceReadableCollection> unspecified

**Errors**

none



Return the name of a class.

**Definition:** <classDescription>







#### **5.4.1 Protocol: <valuable>**

##### **Conforms To**

<Object>

##### **Description**

This protocol describes the behavior for objects that can be evaluated using variants of the











Answers the *value*

value:value:

#### 5.4.6.1 Message Refinement: argumentCount

##### Synopsis

Answers the number of arguments needed to evaluate the receiver.

##### Definition: <valuable>

The number of arguments needed to evaluate the receiver is returned.

##### Refinement: <dyadicValuable>

Returns 2.

##### Return Value

<integer> unspecified

##### Errors

none

#### 5.4.6.2 Message: value: argument1 value: argument2

##### Synopsis

Answers the value of the receiver when applied to the arguments.

##### Definition: <dyadic-valuable>

## **5.5 Exception Protocols**

This section includes the protocols that define the behavior of the exception handling system.



<boolean>      unspecified

**Errors**

none















exception <exceptionDescription> unspecified  
**Return Value**  
<boolean> unspecified



















**Standard Globals**

ZeroDivide

Unspecified language element type. Conforms to the protocol  
<ZeroDivide class>.

**Messages**

dividend:  
*signal*

**5.5.15.1 Message: dividend: argument**

**Synopsis**

Sign of the 203(y)569(is) 8 12 is tegd, Zeo Die (32(c)-8(2))T->F82105jio the58 -1.5542 TD[(Ref the(3(i)-8.3(n)-5.6eanc)  
xct thevnls (of)-1323( )]TJ/F1 1 Tf5.975904 TD0.0024 Tc0 Tw(arum(enf)Tj/TT2 1 Tf2481933 0 TD-0.0048 Tc-0.

uv32.53(v32.ou)-058( tategn)119T4 1.7(r)-6d-13iteg eis 1 8(o theeu)-058( tateg1.7(r)-65.6eateg1.7(r)-68(



**Definition: <exceptionDescription>**

This message is used to determine whether the receiver is a *resumable*





**5.5.19 Protocol: <exceptionSet>**

**Conforms To:**

<exceptionSelector>





**5.6.1.3 Message:**    **> operand**

**Synopsis**

Answer *true*







representations a conversion to a common *numeric representation* is performed, as specified by the Default Conversion Table, before applying the operation. If the resulting protocol is <integer>, then the result value is defined by the ISO/IEC 10967 operation *add<sub>i</sub>*. If the resulting protocol is <Float>, then the result value is defined by the ISO/IEC 10967 *add<sub>f</sub>*. Otherwise, the result is consistent with the mathematical definition of the ISO/IEC 10967 operation *add*.

The protocol and representation of the return value is defined by the Default Result Type. If the return value conforms to <scaledDecimal> then the scale of the result is at least the scale of the receiver after conversion if necessary. If the result value is outside of the



**Definition: <magnitude>**

Answer *true* if the receiver is less than operand with respect to the ordering defined for them.

Answer *false* otherwise.

It is erroneous if the receiver and operand are not *det mpé n.ôa he 0)D(2725-0.0030 000 151D(0020.004V(t is)E*









**Return Values**

<integer> unspecified

**Errors**

none

**5.6.2.19 Message: floor**

**Synopsis**





<number> unspecified

**Errors**



**5.6.2.33 Message: sign**

**Synopsis**

Answer the sign of the receiver.

**Definition: <number>**



Evaluate





**Definition: <rational>**

Treating the receiver as a fraction, answer the lowest common denominator of the receiver.





















**Errors**

none

**5.6.7.6 Message: degreesToRadians**

**Synopsis**

Answer the receiver converted from degrees to radians.











The minimum normalized value allowed by the characterized floating point object representation.









**Return Values**

<Bag> unspecified

**Errors**







Answer the final result of evaluating `operation`







<Object> state

**Errors**

none





**5.7.2.11 Message: keysAndValuesDo: operation**

**Synopsis**

Iteratively evaluate `operation` with each of the receiver's `keys` and values.

**Definition: <abstractDictionary>**

For each *element* in the receiver, `operation` is evaluated with the corresponding *key* as the first







**Description**





**Refinement: <Bag>**

#### **5.7.6.4 Message Refinement: collect: transformer**

##### **Synopsis**

Answer a new collection constructed by gathering the results of evaluating



Duplicates will not be added.

The results are undefined if `newElements` contains *nil*.

**Parameters**

`transformer`



3. For all indices of the receiver, the *element* in the receiver at a given index is *equivalent* to the *element* in *operand* at the same index.

*Element* lookup is defined by the #at : message for the receiver and *operand*.

**Parameters**

*comparand*







**Return Values**

<RECEIVER> new

**Errors**

The replacementElement is not suitable for storage in instance 1.55 57t 1.55 iso fheor57t 1-12.3iv-12(57tor)-6.5'is57



<integer>unspecified

**Errors**

If an evaluation of `discriminator`

**Return Values**

UNSPECIFIED

**Errors**

If the





**5.7.8.26 Message: reverse**























targetElements  
replacementElements

<sequenceReadableCollection> uncaptured  
<sequenceReadableCollection> unspecified

**Return Values**

<readableString> new

**Errors**

none

**5.7.10.11 Message Refinement:**

copyRepl9( )Tw(repl.(5.1(e)2a(l9( )9.1(r)12.m(:)-17 s8(:)-17ar8(:)-17 t9( )-251(p







**5.7.12.1 Message: at: index put: newElement**

**Synopsis**

Replace the *element* in the receiver at `index` with





### **5.7.13.1 Message Refinement: asString**

#### **Synopsis**

Answer a string containing the same characters as the receiver.

#### **Definition: <readableString>**

Answer a string containing the same characters as the receiver, in their original order.

#### **Refinement: <String>**

**5.7.16 Protocol: <sequencedContractibleCollection>**

**Conforms To**  
<collection>

**Description**  
Provides protocol for removing *elements*



operand <sequencedReadableCollection>      uncaptured

**Return Values**

<SortedCollection>      new

**Errors**

If the *elements* of operand cannot be sorted using receiver's *sort block*.

**5.7.17.2 Message Refinement:      add: newElement**

**Synopsis**

Add newElement to the receiver's *elements*.

**Definition: <extensibleCollection>**

This message adds a newElement to the receiver. Unless specifically refined, the position of the newElement in the *element* traversal order is unspecified.

Conformant protocols may place restrictions on the typ4 -1.c6 ita-8.1( on ac)-8.1(e2.2(y)v m)-24.5iel(no)-12.2.(c)-8

The elements are traversed in the order specified by the #do: message for the receiver.

Unless specifically refined, this message is defined to answer an objects conforming to the same protocol as the receiver.

**Refinement: <SortedCollection>**

Answer a <sequencedCollection>.

**Parameters**

transformer <monadicValuable>    uncaptured

**Return Values**

















**Errors**  
none



negative, but it must be non-zero. The *elements*











**Standard Globals**

*Array*

Conforms to the protocol <Array factory>. Its language element type is

count<0

**5.7.24.3 Message Refinement: with: element1**

**Message Refinement: with: element1 with: element2**

**Message Refinement: with: element1 with: element2 with: element3**

**Message Refinement: with: element1 with: element2 with: element3 with: element4**

**Synopsis**

Create a collection initially containing the argument elements



Return a new collection that has space for at least `count` *elements*.

Conforming protocols may refine this message. In particular, the effect of the parameter `count` should be specified in refinements. It can be used to specify the exact number of *elements*, the minimum number, or in some cases can even be interpreted as a hint from the programmer, with no guarantee that the requested number of instance variables will actually be allocated.

Unlesca,







*with:*  
*with:with:*  
*with:with:with*  
*with:with:with:with:*  
*withAll:*

#### **5.7.27.1 Message Refinement: new**

##### **Synopsis**

Create a n







**Return Values**

<Set> new

**Errors**

If any of the elements of `newElements` do not meet the *element type* constraints of the result object







minimum number, or in some cases can even be interpreted as a hint from the programmer, with

If the elements of `newElements`





operand            <Duration>      uncaptured  
**Return Values**

#### **5.8.1.4 Message Refinement: = comparand**

##### **Synopsis**





Answer the number of the day in the year, in the *local time* of the receiver, which includes the





It is an invariant that if x is a <Duration> in range then  
(<DateAndTime> offset: x) offset = x

**Parameters**



**5.8.1.28 Message: year**

**Synopsis**

Answer the number of the year in the *local time*

















`dayOfMonth` greater than the number of days in the month `month` of year `year` of the astronomical Gregorian calendar.

`hour` is not between 0 and 23 inclusive.

`minute` is not between 0 and 59 inclusive.

`second` is not greater than or equal to 0 and strictly less than 60.

the time specified does not exist.



**Definition: <DateAndTime factory>**

N

C



















**Errors**

none

**5.9.4.3 Message:   nextPut: anObject**

**Synopsis**

It is erroneous if the space character is an object that does not conform to the receiver's *sequence*

~~va~~ ~~ty~~ ~~pe~~ ~~2759~~ ~~0~~ ~~9206~~ ~~0~~ ~~8809~~ ~~0~~ ~~8~~ ~~12378~~ ~~0~~ ~~99~~ ~~63~~ ~~0~~ ~~096~~ ~~104~~ ~~are~~ ~~sag~~ ~~ts~~ ~~Synopsis~~ ~~2759~~ ~~0~~ ~~9206~~ ~~0~~ ~~64~~ ~~209~~









## **5.10 File Stream Protocols**





**Messages**

*next:*  
*upTo:*



aString<String>      unspecified

**Return Value**

<readFileStream>      new

**Errors**

As defined by <FileStream factory> #read:type:





## **6. Glossary**

The Smalltalk standard defines and uses the following terms:

*abnormal termination*



cause methods inherited from the class to malfunction. Implementation may use underscore prefixed method selectors or other implementation specific means to implement classes in a non-fragile manner.

*future sequence values* The *sequence values* yet to be read by a stream.

*general subclass* Any class that either directly or indirectly inherits from a superclass is a general subclass of the superclass.

*handle* An *exception handler*



*resumption value*

The value that is returned to the signaler from the *exception action* of a *resumable* exception.

*scope*

*write-back stream*

A stream that supports the writing of objects and that has a *stream backing store* ~~that~~ receives the objects written to the stream. A buffer



sequencedCollection, 207  
sequencedContractibleCollection, 212  
sequencedReadableCollection, 180  
sequencedStream, 274  
Set, 178  
Set factory, 241  
signaledException, 92

## **8. References**