sage: is_GapElement(2)
False

reduce_load()
Returns an invalid GAP element. Note that this is the object returned when a GAP element is unpickled.

EXAMPLES:

```
sage: from sage.interfaces.gap import reduce_load
sage: reduce_load()
...
ValueError: The session in which this object was defined is no longer running.
sage: loads(dumps(gap(2)))
...
ValueError: The session in which this object was defined is no longer running.
```

reduce_load_GAP()
Returns the GAP interface object defined in sage.interfaces.gap.

EXAMPLES:

```
sage: from sage.interfaces.gap import reduce_load_GAP
sage: reduce_load_GAP()
Gap
```

## 14.4 Interface to GAP3

This module implements an interface to GAP3.

AUTHORS:

- Franco Saliola (February 2010)

Warning: GAP3 is not distributed with Sage. You need to install it yourself on your computer. See the section Obtaining GAP3 below.

### 14.4.1 Obtaining GAP3

The GAP3 interface will only work if GAP3 is installed on your computer. You must download and install GAP3 yourself because it is not distributed with Sage. Here are some ways to obtain GAP3:

- download GAP3 from the GAP website:
  

- Jean Michel maintains a version of GAP3 pre-packaged with CHEVIE and VKCURVE:
  

- Frank Luebeck maintains a GAP3 Linux executable, optimized for i686 and statically linked for jobs of 2 GByte or more:
  
### 14.4.2 Changing which GAP3 is used

**Warning:** There is a bug in the pexpect module that prevents you from specifying the path to the GAP3 executable on your system. For now, just make sure that gap3 is in your PATH.

Sage assumes that you can start GAP3 with the command `gap3`; that is, Sage assumes that the command `gap3` is in your PATH. If this is not the case, then you can start GAP3 using the following command:

```python
sage: gap3 = Gap3(command='/usr/local/bin/gap3')
```

### 14.4.3 Functionality and Examples

The interface offers the following functionality.

1. `gap3(expr)` - Evaluation of arbitrary GAP3 expressions, with the result returned as a Sage object wrapping the corresponding GAP3 element:

   ```python
   sage: a = gap3('3+2')
   sage: a
   5
   sage: type(a)
   <class 'sage.interfaces.gap3.GAP3Element'>
   sage: S5 = gap3('SymmetricGroup(5)')
   sage: S5
   Group( (1,5), (2,5), (3,5), (4,5) )
   sage: type(S5)
   <class 'sage.interfaces.gap3.GAP3Record'>
   ```

   This provides a Pythonic interface to GAP3. If `gap_function` is the name of a GAP3 function, then the syntax `gap_element.gap_function()` returns the `gap_element` obtained by evaluating `gap_function(gap_element)` in GAP3:

   ```python
   sage: S5.Size()
   120
   sage: S5.CharTable()
   CharTable( Group( (1,5), (2,5), (3,5), (4,5) ) )
   ```

   Alternatively, you can instead use the syntax `gap3.gap_function(gap_element)`:

   ```python
   sage: gap3.DerivedSeries(S5)
   [ Group( (1,5), (2,5), (3,5), (4,5) ),
     Subgroup( Group( (1,5), (2,5), (3,5), (4,5) ),
              [ (1,2,5), (1,3,5), (1,4,5) ] ) ]
   ```

   If `gap_element` corresponds to a GAP3 record, then `gap_element.recfield` access the record element corresponding to the field `recfield`:

   ```python
   sage: S5.IsRec()
   True
   sage: S5.recfields()
   ['isDomain', 'isGroup', 'identity', 'generators', 'operations',
    'isPermGroup', 'isFinite', '1', '2', '3', '4', 'degree']
   ```
2. By typing `%gap3` or `gap3.interact()` at the command-line, you can interact directly with the underlying GAP3 session.

```python
gap3.interact()
```

3. You can start a new GAP3 session as follows:

```python
gap3.console()
```

4. The interface also has access to the GAP3 help system:

```python
gap3.help('help', pager=False)
```
14.4.4 Common Pitfalls

1. If you want to pass a string to GAP3, then you need to do the following:

```
sage: gap3("This is a GAP3 string")   #optional - gap3
"This is a GAP3 string"
```

This is particularly important when a GAP3 package is loaded via the \texttt{RequirePackage} method (note that one can instead use the \texttt{load_package} method):

```
sage: gap3.RequirePackage("chevie")   #optional - gap3
Welcome to the CHEVIE package, ...
```

14.4.5 Examples

Load a GAP3 package:

```
sage: gap3.load_package("chevie")   #optional - gap3
sage: gap3.version() # random #optional - gap3
'lib: v3r4p4 1997/04/18, src: v3r4p0 1994/07/10, sys: usg gcc ansi'
```

Working with GAP3 lists. Note that GAP3 lists are 1-indexed:

```
sage: L = gap3([[1,2],[3,4]])   #optional - gap3
sage: L[1]   #optional - gap3
1
2
sage: 3 in L   #optional - gap3
True
sage: 4 in L   #optional - gap3
False
sage: m = gap3([[1,2],[3,4]])   #optional - gap3
sage: m[2,1]   #optional - gap3
3
sage: [1,2] in m   #optional - gap3
True
sage: [3,2] in m   #optional - gap3
False
sage: gap3([[1,2]]) in m   #optional - gap3
True
```

Controlling variable names used by GAP3:

```
sage: gap3('2', name='x')   #optional - gap3
2
sage: gap3('x')   #optional - gap3
2
sage: gap3.unbind('x')   #optional - gap3
sage: gap3('x')   #optional - gap3
...   #optional - gap3
TypeError: Gap3 produced error output
Error, Variable: 'x' must have a value...
```
class GAP3Element (parent, value, is_name=False, name=None)

A GAP3 element

Note: If the corresponding GAP3 element is a GAP3 record, then the class is changed to a GAP3Record.

INPUT:

• parent – the GAP session
• value – the GAP3 command as a string
• is_name – bool (default: False); if True, then value is the variable name for the object
• name – str (default: None); the variable name to use for the object. If None, then a variable name is generated.

Note: If you pass E, X or Z for name, then an error is raised because these are sacred variable names in GAP3 that should never be redefined. Sage raises an error because GAP3 does not!

EXAMPLES:

sage: from sage.interfaces.gap3 import GAP3Element #optional - gap3
sage: gap3 = Gap3() #optional - gap3
sage: GAP3Element(gap3, value='3+2') #optional - gap3
5
sage: GAP3Element(gap3, value='sage0', is_name=True) #optional - gap3
5

TESTS:

sage: GAP3Element(gap3, value='3+2', is_name=False, name='X') #optional - gap3
...
ValueError: you are attempting to redefine X; but you should never redefine E, X or Z in gap3 (because

AUTHORS:

• Franco Saliola (Feb 2010)

class GAP3Record (parent, value, is_name=False, name=None)

A GAP3 record

Note: This class should not be called directly, use GAP3Element instead. If the corresponding GAP3 element is a GAP3 record, then the class is changed to a GAP3Record.

AUTHORS:

• Franco Saliola (Feb 2010)

operations()

Return a list of the GAP3 operations for the record.

OUTPUT:

• list of strings - operations of the record

EXAMPLES:

sage: S5 = gap3.SymmetricGroup(5) #optional - gap3
sage: S5.operations() #optional - gap3
[...,'NormalClosure', 'NormalIntersection', 'Normalizer',
 'NumberConjugacyClasses', 'PCore', 'Radical', 'SylowSubgroup',
 'TrivialSubgroup', 'FusionConjugacyClasses', 'DerivedSeries', ...]
sage: S5.DerivedSeries() #optional - gap3
[ Group( (1,5), (2,5), (3,5), (4,5) ),
    Subgroup( Group( (1,5), (2,5), (3,5), (4,5) ),
        [ (1,2,5), (1,3,5), (1,4,5) ] ) ]

recfields()

Return a list of the fields for the record. (Record fields are akin to object attributes in Sage.)

OUTPUT:

• list of strings - the field records

EXAMPLES:

    sage: S5 = gap3.SymmetricGroup(5)                      #optional - gap3
    sage: S5.recfields()                                    #optional - gap3
    ['isDomain', 'isGroup', 'identity', 'generators',
     'operations', 'isPermGroup', 'isFinite', '1', '2',
     '3', '4', 'degree']
    sage: S5.degree                                         #optional - gap3
    5

trait_names()

Defines the list of methods appearing for tab completion.

OUTPUT:

• list of strings – the available fields and operations of the record

EXAMPLES:

    sage: S5 = gap3.SymmetricGroup(5)                      #optional - gap3
    sage: S5.trait_names()                                  #optional - gap3
    ['ConjugacyClassesTry', 'ConjugateSubgroup', 'ConjugateSubgroups',
     'Core', 'DegreeOperation', 'DerivedSeries', 'DerivedSubgroup',
     'Difference', 'DimensionsLoewyFactors', 'DirectProduct', ...]

class Gap3(command='gap3')

A simple Expect interface to GAP3.

EXAMPLES:

    sage: from sage.interfaces.gap3 import Gap3
    sage: gap3 = Gap3(command='gap3')

TESTS:

    sage: gap3(2) == gap3(3)                                 #optional - gap3
    False
    sage: gap3(2) == gap3(2)                                 #optional - gap3
    True
    sage: gap3.trait_names()                                 #optional - gap3
    []

We test the interface behaves correctly after a keyboard interrupt:

    sage: gap3(2)                                              #optional - gap3
    2
    sage: try:                                                
    ... gap3._keyboard_interrupt()
... except:
... pass  #optional - gap3
Interrupting Gap3...
sage: gap3(2)  #optional - gap3
2

We test that the interface busts out of GAP3’s break loop correctly:

sage: f = gap3(\texttt{function(L) return L[0]; end;;})  #optional - gap3
sage: f([1,2,3])  #optional - gap3
...
RuntimeError: Gap3 produced error output
Error, List Element: <position> must be a positive integer at
return L[0] ...

AUTHORS:

• Franco Saliola (Feb 2010)

\texttt{console}()

Spawn a new GAP3 command-line session.

EXAMPLES:

\texttt{sage: gap3.\texttt{console}()}  #not tested

\begin{verbatim}
% Lehrstuhl D fuer Mathematik
% RWTH Aachen
#
#
# Version 3
#
# Release 4.4
#
# 18 Apr 97
#
#
# Alice Niemeyer, Werner Nickel, Martin Schoenert
# Johannes Meier, Alex Wegner, Thomas Bischops
# Frank Celler, Juergen Mnich, Udo Polis
# Thomas Breuer, Goetz Pfeiffer, Hans U. Besche
# Volkmar Felsch, Heiko Theissen, Alexander Hulpke
# Ansgar Kaup, Akos Seress, Erzsebet Horvath
# Bettina Eick
% For help enter: ?<return>

gap>
\end{verbatim}

\texttt{cputime} (t=None)

Returns the amount of CPU time that the GAP session has used in seconds. If \texttt{t} is not None, then it returns the difference between the current CPU time and \texttt{t}.

EXAMPLES:

\texttt{sage: t = gap3.cputime()}  #optional - gap3
\texttt{sage: t} #random  #optional - gap3
0.02
120
sage: gap3.cputime()  #random  #optional - gap3
0.14999999999999999
sage: gap3.cputime(t)  #random  #optional - gap3
0.13

**help**(topic, pager=True)
Print help on the given topic.

**INPUT:**

- topic – string

**EXAMPLES:**

```python
sage: gap3.help('help', pager=False)  #optional - gap3
Help _______________________________________________________
<BLANKLINE>
This section describes together with the following sectio...
help system. The help system lets you read the manual inter...

sage: gap3.help('SymmetricGroup', pager=False)  #optional - gap3
no section with this name was found
```

**TESTS:**

```python
sage: m = gap3([[1,2,3],[4,5,6]]); m  #optional - gap3
[ [ 1, 2, 3 ], [ 4, 5, 6 ] ]
sage: gap3.help('help', pager=False)  #optional - gap3
Help _______________________________________________________
<BLANKLINE>
sage: m  #optional - gap3
[ [ 1, 2, 3 ], [ 4, 5, 6 ] ]
sage: m.Print()  #optional - gap3
[ [ 1, 2, 3 ], [ 4, 5, 6 ] ]
sage: gap3.help('Group', pager=False)  #optional - gap3
Group _____________________________________________________
sage: m  #optional - gap3
[ [ 1, 2, 3 ], [ 4, 5, 6 ] ]
sage: m.Print()  #optional - gap3
[ [ 1, 2, 3 ], [ 4, 5, 6 ] ]
```

**gap3_console**()

Spawn a new GAP3 command-line session.

**EXAMPLES:**

```python
sage: gap3.console()  #not tested
```

14.4. Interface to GAP3
gap3_version()

Return the version of GAP3 that you have in your PATH on your computer.

EXAMPLES:

sage: gap3_version()  # random, optional - gap3
'lib: v3r4p4 1997/04/18, src: v3r4p0 1994/07/10, sys: usg gcc ansi'

14.5 Interface to GP/Pari

Type gp.[tab] for a list of all the functions available from your Gp install. Type gp.[tab]? for Gp’s help about a given function. Type gp(...) to create a new Gp object, and gp.eval(...) to run a string using Gp (and get the result back as a string).

EXAMPLES: We illustrate objects that wrap GP objects (gp is the PARI interpreter):

sage: M = gp('[1,2;3,4]')
sage: M
[1, 2; 3, 4]
sage: M * M
[7, 10; 15, 22]
sage: M + M
[2, 4; 6, 8]
sage: M.matdet()  
-2

sage: E = gp.ellinit([1,2,3,4,5])
sage: E.ellglobalred()
[10351, [1, -1, 0, -1], 1]
sage: E.ellan(20)
[1, 1, 0, -1, -3, 0, -1, -3, -3, -1, 0, 1, -1, 0, -1, 5, -3, 4, 3]

sage: primitive_root(7)
3
sage: x = gp("znlog( Mod(2,7), Mod(3,7))")
sage: 3^x % 7
2