
Sage 9.5 Reference Manual: Quasimodular Forms

Release 9.5

The Sage Development Team

Jan 31, 2022

CONTENTS

1	Module List	1
1.1	Graded quasimodular forms ring	1
1.2	Elements of quasimodular forms rings	7
2	Indices and Tables	13
	Python Module Index	15
	Index	17

MODULE LIST

1.1 Graded quasimodular forms ring

Let E_2 be the weight 2 Eisenstein series defined by

$$E_2(z) = 1 - \frac{2k}{B_k} \sum_{n=1}^{\infty} \sigma(n)q^n$$

where σ is the sum of divisors function and $q = \exp(2\pi iz)$ is the classical parameter at infinity, with $\text{im}(z) > 0$. This weight 2 Eisenstein series is not a modular form as it does not satisfy the modularity condition:

$$z^2 E_2(-1/z) = E_2(z) + \frac{2k}{4\pi i B_k z}.$$

E_2 is a quasimodular form of weight 2. General quasimodular forms of given weight can also be defined. We denote by QM the graded ring of quasimodular forms for the full modular group $SL_2(\mathbf{Z})$.

The SageMath implementation of the graded ring of quasimodular forms uses the following isomorphism:

$$QM \cong M_*[E_2]$$

where $M_* \cong \mathbf{C}[E_4, E_6]$ is the graded ring of modular forms for $SL_2(\mathbf{Z})$. (see `sage.modular.modform.ring.ModularFormRing()`).

EXAMPLES:

```
sage: QM = QuasiModularForms(1); QM
Ring of Quasimodular Forms for Modular Group SL(2,Z) over Rational Field
sage: QM.gens()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6),
1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + O(q^6),
1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + O(q^6)]
sage: E2 = QM.0; E4 = QM.1; E6 = QM.2
sage: E2 * E4 + E6
2 - 288*q - 20304*q^2 - 185472*q^3 - 855216*q^4 - 2697408*q^5 + O(q^6)
sage: E2.parent()
Ring of Quasimodular Forms for Modular Group SL(2,Z) over Rational Field
```

REFERENCE:

See section 5.3 (page 58) of [Zag2008]

AUTHORS:

- David Ayotte (2021-03-18): initial version

```
class sage.modular.quasimodform.ring.QuasiModularForms(group=1, base_ring=Rational Field,
                                                       name='E2')
    Bases: sage.structure.parent.Parent, sage.structure.unique_representation.UniqueRepresentation
```

The graded ring of quasimodular forms for the full modular group $SL_2(\mathbf{Z})$, with coefficients in a ring.

EXAMPLES:

```
sage: QM = QuasiModularForms(1); QM
Ring of Quasimodular Forms for Modular Group SL(2,Z) over Rational Field
sage: QM.gens()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6),
 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + O(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + O(q^6)]
```

It is possible to access the weight 2 Eisenstein series:

```
sage: QM.weight_2_eisenstein_series()
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6)
```

Currently, the only supported base ring is the rational numbers:

```
sage: QuasiModularForms(1, GF(5))
Traceback (most recent call last):
...
NotImplementedError: base ring other than Q are not yet supported for quasimodular_
->forms ring
```

Element

alias of `sage.modular.quasimodform.element.QuasiModularFormsElement`

from_polynomial(*polynomial*)

Convert the given polynomial $P(X, Y, Z)$ to the graded quasiform $P(E_2, E_4, E_6)$ where E_2 , E_4 and E_6 are the generators given by `gens()`.

INPUT:

- `polynomial` – A multivariate polynomial

OUTPUT: the graded quasimodular forms $P(E_2, E_4, E_6)$

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: P.<x, y, z> = QQ[]
sage: QM.from_polynomial(x)
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6)
sage: QM.from_polynomial(x) == QM.0
True
sage: QM.from_polynomial(y) == QM.1
True
sage: QM.from_polynomial(z) == QM.2
True
sage: QM.from_polynomial(x^2 + y + x*z + 1)
4 - 336*q - 2016*q^2 + 322368*q^3 + 3691392*q^4 + 21797280*q^5 + O(q^6)
```

gen(*n*)

Return the *n*-th generator of the quasimodular forms ring.

EXAMPLES:

```

sage: QM = QuasiModularForms(1)
sage: QM.0
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6)
sage: QM.1
1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6)
sage: QM.2
1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)
sage: QM = QuasiModularForms(5)
sage: QM.0
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6)
sage: QM.1
1 + 6*q + 18*q^2 + 24*q^3 + 42*q^4 + 6*q^5 + 0(q^6)
sage: QM.2
1 + 240*q^5 + 0(q^6)
sage: QM.3
q + 10*q^3 + 28*q^4 + 35*q^5 + 0(q^6)
sage: QM.4
Traceback (most recent call last):
...
IndexError: list index out of range

```

generators()

Return a list of generators of the quasimodular forms ring. Note that the generators of the modular forms subring are the one given by the method :meth: `sage.modular.modform.ring.ModularFormsRing.gen_forms`

EXAMPLES:

```

sage: QM = QuasiModularForms(1)
sage: QM.gens()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6),
 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)]
sage: QM.modular_forms_subring().gen_forms()
[1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)]
sage: QM = QuasiModularForms(5)
sage: QM.gens()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6),
 1 + 6*q + 18*q^2 + 24*q^3 + 42*q^4 + 6*q^5 + 0(q^6),
 1 + 240*q^5 + 0(q^6),
 q + 10*q^3 + 28*q^4 + 35*q^5 + 0(q^6)]

```

An alias of this method is `generators`:

```

sage: QuasiModularForms(1).generators()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6),
 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)]

```

gens()

Return a list of generators of the quasimodular forms ring. Note that the generators of the modular forms subring are the one given by the method :meth: `sage.modular.modform.ring.ModularFormsRing.gen_forms`

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.gens()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6),
 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)]
sage: QM.modular_forms_subring().gen_forms()
[1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)]
sage: QM = QuasiModularForms(5)
sage: QM.gens()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6),
 1 + 6*q + 18*q^2 + 24*q^3 + 42*q^4 + 6*q^5 + 0(q^6),
 1 + 240*q^5 + 0(q^6),
 q + 10*q^3 + 28*q^4 + 35*q^5 + 0(q^6)]
```

An alias of this method is `generators`:

```
sage: QuasiModularForms(1).generators()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6),
 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)]
```

group()

Return the congruence subgroup for which this is the ring of quasimodular forms.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.group()
Modular Group SL(2,Z)
sage: QM.group() is SL2Z
True
sage: QuasiModularForms(3).group()
Congruence Subgroup Gamma0(3)
sage: QuasiModularForms(Gamma1(5)).group()
Congruence Subgroup Gamma1(5)
```

modular_forms_of_weight(weight)

Return the space of modular forms on this group of the given weight.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.modular_forms_of_weight(12)
Modular Forms space of dimension 2 for Modular Group SL(2,Z) of weight 12 over_
↳Rational Field
sage: QM = QuasiModularForms(Gamma1(3))
sage: QM.modular_forms_of_weight(4)
Modular Forms space of dimension 2 for Congruence Subgroup Gamma1(3) of weight_
↳4 over Rational Field
```

(continues on next page)

(continued from previous page)

modular_forms_subring()

Return the subring of modular forms of this ring of quasimodular forms.

EXAMPLES:

```
sage: QuasiModularForms(1).modular_forms_subring()
Ring of Modular Forms for Modular Group SL(2,Z) over Rational Field
sage: QuasiModularForms(5).modular_forms_subring()
Ring of Modular Forms for Congruence Subgroup Gamma0(5) over Rational Field
```

ngens()

Return the number of generators of the given graded quasimodular forms ring.

EXAMPLES:

```
sage: QuasiModularForms(1).ngens()
3
```

one()

Return the one element of this ring.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.one()
1
sage: QM.one().is_one()
True
```

polygen()

Return the generator of this quasimodular form space as a polynomial ring over the modular form subring. Note that this generator correspond to the weight-2 Eisenstein series. The default name of this generator is 'E2'.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.polygen()
E2
sage: QuasiModularForms(1, name='X').polygen()
X
sage: QM.polygen().parent()
Univariate Polynomial Ring in E2 over Ring of Modular Forms for Modular Group
↪SL(2,Z) over Rational Field
```

polynomial_ring(names='E2, E4, E6')

Return a multivariate polynomial ring isomorphic to the given graded quasimodular forms ring.

In the case of the full modular group, this ring is $R[E_2, E_4, E_6]$ where E_2 , E_4 and E_6 have degrees 2, 4 and 6 respectively.

INPUT:

- names (str, default: 'E2, E4, E6') – a list or tuple of names (strings), or a comma separated string. Correspond to the names of the variables.

OUTPUT: A multivariate polynomial ring in the variables names

EXAMPLES:

```
sage: QM = QuasiModularForms(1) sage: P.<E2, E4, E6> = QM.polynomial_ring(); P Multi-
variate Polynomial Ring in E2, E4, E6 over Rational Field sage: E2.degree() 2 sage: E4.degree()
4 sage: E6.degree() 6 sage: P.<x, y, z, w> = QQ[] sage: QM.from_polynomial(x+y+z+w) Trace-
back (most recent call last): ... ValueError: the number of variables (4) of the given polynomial
cannot exceed the number of generators (3) of the quasimodular forms ring
```

quasimodular_forms_of_weight(*weight*)

Return the space of quasimodular forms on this group of the given weight.

INPUT:

- *weight* (int, Integer)

OUTPUT: A quasimodular forms space of the given weight.

EXAMPLES:

```
sage: QuasiModularForms(1).quasimodular_forms_of_weight(4)
Traceback (most recent call last):
...
NotImplementedError: spaces of quasimodular forms of fixed weight not yet
↳ implemented
```

some_elements()

Return a list of generators of self.

EXAMPLES:

```
sage: QuasiModularForms(1).some_elements()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6),
 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + 0(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + 0(q^6)]
```

weight_2_eisenstein_series()

Return the weight 2 Eisenstein series.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: E2 = QM.weight_2_eisenstein_series(); E2
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + 0(q^6)
sage: E2.parent()
Ring of Quasimodular Forms for Modular Group SL(2,Z) over Rational Field
```

zero()

Return the zero element of this ring.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.zero()
0
sage: QM.zero().is_zero()
True
```

1.2 Elements of quasimodular forms rings

AUTHORS:

- DAVID AYOTTE (2021-03-18): initial version

class `sage.modular.quasimodform.element.QuasiModularFormsElement` (*parent, polynomial*)

Bases: `sage.structure.element.ModuleElement`

A quasimodular forms ring element. Such an element is described by SageMath as a polynomial

$$f_0 + f_1 E_2 + f_2 E_2^2 + \cdots + f_m E_2^m$$

where each f_i a graded modular form element (see `GradedModularFormElement`)

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.gens()
[1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6),
 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + O(q^6),
 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + O(q^6)]
sage: QM.0 + QM.1
2 + 216*q + 2088*q^2 + 6624*q^3 + 17352*q^4 + 30096*q^5 + O(q^6)
sage: QM.0 * QM.1
1 + 216*q - 3672*q^2 - 62496*q^3 - 322488*q^4 - 1121904*q^5 + O(q^6)
sage: (QM.0)^2
1 - 48*q + 432*q^2 + 3264*q^3 + 9456*q^4 + 21600*q^5 + O(q^6)
sage: QM.0 == QM.1
False
```

Quasimodular forms ring element can be created via a polynomial in E_2 over the ring of modular forms:

```
sage: E2 = QM.polygen()
sage: E2.parent()
Univariate Polynomial Ring in E2 over Ring of Modular Forms for Modular Group SL(2,
↔Z) over Rational Field
sage: QM(E2)
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6)
sage: M = QM.modular_forms_subring()
sage: QM(M.0 * E2 + M.1 * E2^2)
2 - 336*q + 4320*q^2 + 398400*q^3 - 3772992*q^4 - 89283168*q^5 + O(q^6)
```

derivative()

Return the derivative $q \frac{d}{dq}$ of the given quasimodular form.

If the form is not homogeneous, then this method sums the derivative of each homogeneous component.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: E2, E4, E6 = QM.gens()
sage: dE2 = E2.derivative(); dE2
-24*q - 144*q^2 - 288*q^3 - 672*q^4 - 720*q^5 + O(q^6)
sage: dE2 == (E2^2 - E4)/12 # Ramanujan identity
True
```

(continues on next page)

(continued from previous page)

```

sage: dE4 = E4.derivative(); dE4
240*q + 4320*q^2 + 20160*q^3 + 70080*q^4 + 151200*q^5 + O(q^6)
sage: dE4 == (E2 * E4 - E6)/3 # Ramanujan identity
True
sage: dE6 = E6.derivative(); dE6
-504*q - 33264*q^2 - 368928*q^3 - 2130912*q^4 - 7877520*q^5 + O(q^6)
sage: dE6 == (E2 * E6 - E4^2)/2 # Ramanujan identity
True

```

Note that the derivative of a modular form is not necessarily a modular form:

```

sage: dE4.is_modular_form()
False
sage: dE4.weight()
6

```

homogeneous_components()

Return a dictionary where the values are the homogeneous components of the given graded form and the keys are the weights of those components.

EXAMPLES:

```

sage: QM = QuasiModularForms(1)
sage: (QM.0).homogeneous_components()
{2: 1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6)}
sage: (QM.0 + QM.1 + QM.2).homogeneous_components()
{2: 1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6),
4: 1 + 240*q + 2160*q^2 + 6720*q^3 + 17520*q^4 + 30240*q^5 + O(q^6),
6: 1 - 504*q - 16632*q^2 - 122976*q^3 - 532728*q^4 - 1575504*q^5 + O(q^6)}
sage: (1 + QM.0).homogeneous_components()
{0: 1, 2: 1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6)}

```

is_graded_modular_form()

Return True if the given quasimodular form is a graded modular forms element and False otherwise.

EXAMPLES:

```

sage: QM = QuasiModularForms(1)
sage: (QM.0).is_graded_modular_form()
False
sage: (QM.1).is_graded_modular_form()
True
sage: (QM.1 + QM.0^2).is_graded_modular_form()
False
sage: (QM.1^2 + QM.2).is_graded_modular_form()
True
sage: QM = QuasiModularForms(Gamma0(6))
sage: (QM.0).is_graded_modular_form()
False
sage: (QM.1 + QM.2 + QM.1 * QM.3).is_graded_modular_form()
True
sage: QM.zero().is_graded_modular_form()
True

```

Note: A graded modular form in SageMath is not necessarily a modular form as it can have mixed weight components. To check for modular forms only, see the method `is_modular_form()`.

is_homogeneous()

Return True if the graded quasimodular form is a homogeneous element, that is it lives in a unique graded components of the graded ring of self.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: (QM.0).is_homogeneous()
True
sage: (QM.0 + QM.1).is_homogeneous()
False
sage: (QM.0 * QM.1 + QM.2).is_homogeneous()
True
sage: QM(1).is_homogeneous()
True
sage: (1 + QM.0).is_homogeneous()
False
```

is_modular_form()

Return True if the given quasimodular form is a modular form and False otherwise.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: (QM.0).is_modular_form()
False
sage: (QM.1).is_modular_form()
True
sage: (QM.1 + QM.2).is_modular_form() # mixed weight components
False
sage: QM.zero().is_modular_form()
True
```

is_one()

Return “True” if the quasiform is 1 and “False” otherwise

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: QM.one().is_one()
True
sage: QM(1).is_one()
True
sage: (QM.0).is_one()
False
```

is_zero()

Return “True” if the quasiform is 0 and “False” otherwise

EXAMPLES:

```

sage: QM = QuasiModularForms(1)
sage: QM.zero().is_zero()
True
sage: QM(0).is_zero()
True
sage: QM(1/2).is_zero()
False
sage: (QM.0).is_zero()
False

```

polynomial(names='E2, E4, E6')

Return a multivariate polynomial $P(E_2, E_4, E_6)$ corresponding to the given form where E_2 , E_4 and E_6 are the generators of the quasimodular form ring given by `gens()`.

INPUT:

- `names` (str, default: 'E2, E4, E6') – a list or tuple of names (strings), or a comma separated string. Correspond to the names of the variables;

OUTPUT: A multivariate polynomial in the variables `names`

EXAMPLES:

```

sage: QM = QuasiModularForms(1)
sage: (QM.0 + QM.1).polynomial()
E4 + E2
sage: (1/2 + QM.0 + 2*QM.1^2 + QM.0*QM.2).polynomial()
E2*E6 + 2*E4^2 + E2 + 1/2

```

q_expansion(prec=6)

Computes the q -expansion of self to precision `prec`.

An alias of this method is `qexp`.

EXAMPLES:

```

sage: QM = QuasiModularForms()
sage: E2 = QM.0
sage: E2.q_expansion()
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6)
sage: E2.q_expansion(prec=10)
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 - 288*q^6 - 192*q^7 - 360*q^8 -
↪312*q^9 + O(q^10)

```

qexp(prec=6)

Computes the q -expansion of self to precision `prec`.

An alias of this method is `qexp`.

EXAMPLES:

```

sage: QM = QuasiModularForms()
sage: E2 = QM.0
sage: E2.q_expansion()
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 + O(q^6)
sage: E2.q_expansion(prec=10)
1 - 24*q - 72*q^2 - 96*q^3 - 168*q^4 - 144*q^5 - 288*q^6 - 192*q^7 - 360*q^8 -
↪312*q^9 + O(q^10)

```

serre_derivative()

Return the Serre derivative of the given quasimodular form.

If the form is not homogeneous, then this method sums the Serre derivative of each homogeneous component.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: E2, E4, E6 = QM.gens()
sage: DE2 = E2.serre_derivative(); DE2
-1/6 - 16*q - 216*q^2 - 832*q^3 - 2248*q^4 - 4320*q^5 + O(q^6)
sage: DE2 == (-E2^2 - E4)/12
True
sage: DE4 = E4.serre_derivative(); DE4
-1/3 + 168*q + 5544*q^2 + 40992*q^3 + 177576*q^4 + 525168*q^5 + O(q^6)
sage: DE4 == (-1/3) * E6
True
sage: DE6 = E6.serre_derivative(); DE6
-1/2 - 240*q - 30960*q^2 - 525120*q^3 - 3963120*q^4 - 18750240*q^5 + O(q^6)
sage: DE6 == (-1/2) * E4^2
True
```

The Serre derivative raises the weight of homogeneous elements by 2:

```
sage: F = E6 + E4 * E2
sage: F.weight()
6
sage: F.serre_derivative().weight()
8
```

to_polynomial(names='E2, E4, E6')

Return a multivariate polynomial $P(E_2, E_4, E_6)$ corresponding to the given form where E_2 , E_4 and E_6 are the generators of the quasimodular form ring given by `gens()`.

INPUT:

- `names` (str, default: 'E2, E4, E6') – a list or tuple of names (strings), or a comma separated string. Correspond to the names of the variables;

OUTPUT: A multivariate polynomial in the variables `names`

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: (QM.0 + QM.1).polynomial()
E4 + E2
sage: (1/2 + QM.0 + 2*QM.1^2 + QM.0*QM.2).polynomial()
E2*E6 + 2*E4^2 + E2 + 1/2
```

weight()

Return the weight of the given quasimodular form.

Note that the given form must be homogeneous.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: (QM.0).weight()
2
sage: (QM.0 * QM.1 + QM.2).weight()
6
sage: QM(1/2).weight()
0
sage: (QM.0 + QM.1).weight()
Traceback (most recent call last):
...
ValueError: the given graded quasiform is not an homogeneous element
```

weights_list()

Return the list of the weights of all the graded components of the given graded quasimodular form.

EXAMPLES:

```
sage: QM = QuasiModularForms(1)
sage: (QM.0).weights_list()
[2]
sage: (QM.0 + QM.1 + QM.2).weights_list()
[2, 4, 6]
sage: (QM.0 * QM.1 + QM.2).weights_list()
[6]
sage: QM(1/2).weights_list()
[0]
```


INDICES AND TABLES

- [Index](#)
- [Module Index](#)
- [Search Page](#)

PYTHON MODULE INDEX

m

`sage.modular.quasimodform.element`, 7

`sage.modular.quasimodform.ring`, 1

INDEX

D

`derivative()` (*sage.modular.quasimodform.element.QuasiModularFormsElement*, *method*), 7

E

`Element` (*sage.modular.quasimodform.ring.QuasiModularForms* attribute), 2

F

`from_polynomial()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 2

G

`gen()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 2

`generators()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 3

`gens()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 3

`group()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 4

H

`homogeneous_components()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 8

I

`is_graded_modular_form()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 8

`is_homogeneous()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 9

`is_modular_form()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 9

`is_one()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 9

`is_zero()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 9

M

`modular_forms_of_weight()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 4

`modular_forms_subring()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 5

module

`sage.modular.quasimodform.element`, 7

`sage.modular.quasimodform.ring`, 1

N

`ngens()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 5

O

`one()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 5

P

`polygen()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 5

`polynomial()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 10

`polynomial_ring()` (*sage.modular.quasimodform.ring.QuasiModularFormsElement* *method*), 5

Q

`q_expansion()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 10

`q_exp()` (*sage.modular.quasimodform.element.QuasiModularFormsElement* *method*), 10

`quasimodular_forms_of_weight()` (*sage.modular.quasimodform.ring.QuasiModularForms* *method*), 8

`QuasiModularForms` (class in *sage.modular.quasimodform.ring*), 1

`QuasiModularFormsElement` (class in *sage.modular.quasimodform.element*), 7

S

`sage.modular.quasimodform.element`

module, 7
sage.modular.quasimodform.ring
 module, 1
serre_derivative() (*sage.modular.quasimodform.element.QuasiModularFormsElement*
 method), 10
some_elements() (*sage.modular.quasimodform.ring.QuasiModularForms*
 method), 6

T

to_polynomial() (*sage.modular.quasimodform.element.QuasiModularFormsElement*
 method), 11

W

weight() (*sage.modular.quasimodform.element.QuasiModularFormsElement*
 method), 11
weight_2_eisenstein_series()
 (*sage.modular.quasimodform.ring.QuasiModularForms*
 method), 6
weights_list() (*sage.modular.quasimodform.element.QuasiModularFormsElement*
 method), 12

Z

zero() (*sage.modular.quasimodform.ring.QuasiModularForms*
 method), 6