

---

# **clamp Documentation**

***Release 0.1***

**Darjus Loktevic**

March 02, 2017



<b>1</b>	<b>clamp API</b>	<b>3</b>
<b>2</b>	<b>Indices and tables</b>	<b>5</b>
	<b>Python Module Index</b>	<b>7</b>



Contents:



---

## clamp API

---

`clamp.method(return_type, arg_types=[], name=None)`

Decorate a clamp method with Java type information

### Parameters

- **return\_type** (*java.lang.reflect.Type*) – method return type
- **arg\_types** (*array of java.lang.reflect.Type*) – argument types that the method accepts
- **name** (*str*) – alternative method name. You can use this to support polymorphism.

Examples...

Simple method with `Void.TYPE` as the return type:

```
public void returnsVoid()
```

```
@clamp.method(Void.TYPE)
def returnsVoid(self):
    pass
```

This polymorphism example will generate a Java method with the following signature:

```
public int add(int a, int b)
public long add(long a, long b)
```

```
@clamp.method(int.TYPE, [int.TYPE, int.TYPE])
def add(self, a, b):
    return a + b

# we override the method name to "add", which will indicate that we want
# create a method with the same name
@clamp.method(long.TYPE, [long.TYPE, long.TYPE], name="add")
def addLong(self, a, b):
    return a_long + b_long
```

`clamp.throws(*exceptions)`

Declare what Java exceptions this method throws

**Parameters** **exceptions** (*a list of java.lang.Throwable*) – accepts a list of exceptions as arguments

Example:

```
public void writeList() throws IOException, NoSuchFieldException
```

```
@clamp.throws(IOException, NoSuchFieldException)
@clamp.method(Void.TYPE)
def writeList(self):
    raise IOException("message")
    raise NoSuchFieldException("other message")
```

`clamp.annotated(annotation, **kwargs)`

Annotate the Java version of the method with Java annotation

### Parameters

- **annotation** (*java.lang.annotation.Annotation*) – Java annotation
- **kwargs** (*java.lang.Object*) – keyword arguments to pass to the annotation

Example annotation:

```
@junit.Test
public void test()
```

```
@clamp.annotated(Test)
@clamp.method(Void.TYPE)
def test(self):
    pass
```



---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`



**C**

`clamp`, 3



## A

`annotated()` (in module `clamp`), 4

## C

`clamp` (module), 3

## M

`method()` (in module `clamp`), 3

## T

`throws()` (in module `clamp`), 3