Instance methods

An instance method object combines a class, a class instance and any callable object (normally a user-defined function).

Special read-only attributes: `__self__` is the class instance object,`__func__` is the function object; `__doc__` is the method's documentation (same as`__func__.__doc__`);`__name__` is the method name (same as`__func__.__name__`);`__module__` is the name of the module the method was defined in, or `None` if unavailable.

Methods also support accessing (but not setting) the arbitrary function attributes on the underlying function object.

User-defined method objects may be created when getting an attribute of a class (perhaps via an instance of that class), if that attribute is a user-defined function object or a class method object.

When an instance method object is created by retrieving a user-defined function object from a class via one of its instances, its`__self__` attribute is the instance, and the method object is said to be bound. The new method's`__func__` attribute is the original function object.

When a user-defined method object is created by retrieving another method object from a class or instance, the behaviour is the same as for a function object, except that the`__func__` attribute of the new instance is not the original method object but its `__func__` attribute.

When an instance method object is created by retrieving a class method object from a class or instance, its `__self__`attribute is the class itself, and its `__func__` attribute is the function object underlying the class method.

When an instance method object is called, the underlying function (`__func__`) is called, inserting the class instance (`__self__`) in front of the argument list. For instance, when `C` is a class which contains a definition for a function `f()`, and `x` is an instance of `C`, calling`x.f(1)` is equivalent to calling `C.f(x, 1)`.

When an instance method object is derived from a class method object, the "class instance" stored in`__self__` will actually be the class itself, so that calling either `x.f(1)` or`C.f(1)` is equivalent to calling `f(C,1)` where `f` is the underlying function.

Note that the transformation from function object to instance method object happens each time the attribute is retrieved from the instance. In some cases, a fruitful optimization is to assign the attribute to a local variable and call that local variable. Also notice that this transformation only happens for user-defined functions; other callable objects (and all non-callable objects) are retrieved without transformation. It is also important to note that user-defined functions which are attributes of a class instance are not converted to bound methods; this *only*happens when the function is an attribute of the class.