R"""Components

Author : Anders Andersen
Created On : Wed Jul 29 10:49:53 1998
Last Modified By: Anders Andersen
Last Modified On: Fri Apr 7 17:07:58 2000
Status : Unknown, Use with caution!

This module implements a component class (and a component factory). We use this class to create components from existing objects. We can create a component `ca` for the object `a` with the interface `"out"` implemented in the interface reference `a.out` with this Python statement:

```
ca = Component({"out":a.out}, a)
```

The component class contains a reference to the object it is representing (an object graph if it is a composite object [see `composite.py`]) and an interface dictionary containing mappings from interface names to interface references of the object.

The figure above gives an example of two components with one interface each. These interfaces are bound together with a local binding (often drawn as shown on the left part of the figure). The right part of the figure shows the details of the implementation of local bindings between components. A local binding between the two components `ca` and `cb` through the interfaces `"out"` and `"in"` is created with this local bind call:

```
lb = localBind(ca.interfaces["out"], cb.interfaces["in"])
```

"""

```
# Miscellaneous definitions and values common for the Open-ORB core
from misc import *


# Signal interfaces
from lbind import *


class ComponentException(OpenORBException):
```

R"""Component exception

All exceptions or errors introduced by the `component` and `composite` module is handled by this exception class.

"""

```
    pass
```

```
class _eventMethod:

    def __init__(self, iref=None):
        self.iref = iref

    def __call__(self, *args, **kw):
        apply(getattr(self.iref, "event"), (), {})


class Component:
    R"""A component reference for objects

    An object can use this class to become a component. A component is an object with public named
    interfaces.

    """

    def __init__(self, interfaces={}, object=None):
        R"""Register an object with interfaces

        We save the object reference and its named interfaces. The interfaces attribute is a dictionary
        with mappings from the name of an interface to its interface reference. A component with two
        interfaces with interface references iin and iout named "in" and "out" have an interface attribute
        with this value: {"in":iin, "out":iout}.

        """
        self.interfaces = interfaces
        self.object = object
        self.events = {}
        self.notifications = {}

    def event(self, name):
        self.events[name] = SigSrcIRef(self)
        return _eventMethod(self.events[name])

    def notification(self, name, m):
        self.notifications[name] = SigSinkIRef()
        self.notifications[name].__local__["object"] = self
        self.notifications[name].__local__["iobj"] = IObj()
        self.notifications[name].__local__["iobj"].__dict__["event"] = m


def componentFactory(ifaces=[], mkobj=None, args=(), kw={}):
    R"""A factory for components

    Creates a component including an instance of the given object class. The factory expects to find interfaces
    in the objects name space with the same names as the ones given in ifaces.

    """
    object = apply(mkobj, args, kw)
    interfaces = {}
    if type(ifaces) is DictType:
       for ifname in ifaces.keys():
           interfaces[ifname] = ifaces[ifname]
           interfaces[ifname].__local__["object"] = object
           interfaces[ifname].__testExpInterface__(ifaces[ifname].__expID__)
    else:
        for ifname in ifaces:
            interfaces[ifname] = getattr(object, ifname)
    return Component(interfaces, object)
```

```
# LocalWords: Jul Aug UK NORUT aacodefont py hfil pt CB localBind def init cb
# LocalWords: irin irout Oct parbox linewidth includegraphics ifaces mkobj kw
# LocalWords: componentFactory args ifname getattr
```