

JCloneable

The template class JCloneable resides in the name space JLANG and constitutes an auxiliary interface for cloning of objects. It defines the virtual method clone().

A possible implementation of class JCloneable is:

```
template<class JCloneable_t>
struct JCloneable<JCloneable_t>
{
    typedef JCloneable_t* clone_type;

    virtual ~JCloneable()
    {}

    virtual clone_type clone() const = 0;
};
```

To make this work, a class X should simply derive from JCloneable.

```
struct X :
    public JCloneable<X>
{
    virtual void x() const = 0;
};
```

A concrete class that derives from the interface X should then provide an implementation of both virtual methods clone() and x().

The class JCloneable allows for a second template parameter. In that case, an implementation of the virtual method clone() is provided based on the copy constructor of the derived class. In addition, the interface itself (i.e. first template argument) is defined. For example.

```
struct A :
    public JCloneable<A>
{
    virtual int get() const = 0;
};
```

```
struct B :
    public JCloneable<A, B>
{
    static const int value = 1;

    virtual int get() const override
    {
        return value;
    }
};
```

```
B b;
C c;
```

```
A* p[] = { b.clone(), c.clone() };
```

```
struct C :
    public JCloneable<A, C>
{
    static const int value = 2;

    virtual int get() const override
    {
        return value;
    }
};
```

```
cout << p[0]->get() << ' ' << p[1]->get() << endl;
```

```
delete p[0];  
delete p[1];
```

will produce

1 2