```
// Stephen M. Jones // 05/12/2024

// Coda-C example #4:   Virtual Functions

// Task:  Show how a virtual function is called and how to define one.

static void tester(Objobj) {
    char*tag="***NO TAG FUNCTION***";
    if(OResponds(obj,xmlTag)) tag=obj_(xmlTag,obj);
    printf("Object kind:  %-10.10s  xmlTag=%s\n",kindO(obj),tag);
    }

static char *Void_xmlTag(Objobj) { returnOs("Voidy"); }

static void codax04() {
    cleanO Array a=newO(Array); tester(a);
    cleanODict  d=newO(Dict);  tester(d);
    cleanO Char  c=newO(Char);  tester(c);
    cleanO Void  v=alocO(1);    tester(v);

    codac_sig(Class_Void,sig_xmlTag,Void_xmlTag); tester(v);
    codac_sig(Class_Void,sig_xmlTag,0);           tester(v);
    }

codax_register(codax04)

// Purpose:  To demonstrate how virtual functions are called with an object and signature.
```

```
// Coda-C adds the following:

// Obj         - C data type aka (void *)
// OResponds() - determine if a virtual function is defined for an object's class.
// obj_()      - the Coda-C objective call.
// kindO()     - return the class name of an object.
// Os()        - C data type, defines a static string object.  Class: ConstChar
// codac_sig() - register a virtual function.

/*<stdout> example's output
Object kind:  ArrayxmlTag=array
Object kind:  DictionaryxmlTag=dict
Object kind:  CharxmlTag=string
Object kind:  VoidxmlTag=***NO TAG FUNCTION***
Object kind:  VoidxmlTag=Voidy
Object kind:  VoidxmlTag=***NO TAG FUNCTION***
**********************/
```